

FINAL REPORT
"Topography from Shading and Stereo"
Contract Number: NAS5-31352

Final
11-42
1979
32-739
p. 190

Tasks Completed:

- Analyzed and compared competing approaches to 'phototopography'
- Implemented and compared cost functions for topographic recovery
- Implemented planet phototopography from shading and stereo
- Established accuracy through experiments with synthetic images
- Demonstrated that phototopography can yield quantitative information
- Establish robustness through experiments with real planetary image
- Developed recommendations for image taking strategy
- Ported the phototopography implementation from MATLAB to C
- Made the C language implementation available for anonymous FTP.
- Made preliminary effort to apply new approach to other imaging tasks.

(This covers essentially all of the milestones listed in the original contract proposal except for the extension to more than two images which was planned for the third year).

Summary:

Methods exploiting photometric information in images that have been developed in machine vision can be applied to planetary imagery. Older techniques, however, always focused on just one visual cue, such as shading or binocular stereo, and produced results that are either not very accurate in an absolute sense or provide information only at a few points on the surface.

Integrating shape from shading, binocular stereo and photometric stereo yields a robust system for recovering detailed surface shape and surface reflectance information. Such a system is useful in producing quantitative information from the vast volume of imagery being received, as well as in helping visualize the underlying surface. (For additional background information

please refer to the original contract proposal "Topography from Shading and Stereo" by B.K.P. Horn and M. Caplinger.)

Methods for fusing two computer vision methods are discussed in Clay Thompson's thesis and several example algorithms are presented to illustrate the variational method of fusing algorithms. The example algorithms solve the *photo-topography problem*: that is, the algorithms seek to determine planet topography given two images taken from two different locations with two different lighting conditions. The algorithms each employ a single cost function that combines the computer vision methods of shape-from-shading and stereo in different ways. The algorithms are closely coupled and take into account all the constraints of the photo-topography problem.

One such algorithm, the *z-only* algorithm, can accurately and robustly estimate the height of a surface from two given images. Results of running the algorithms on four synthetic test image sets of varying difficulty are presented in Clay Thompson's thesis.

These results are further extended in Charles Yan's thesis, where spatially varying albedo is introduced. Charles Yan also ported Clay Thompson's MATLAB code to C to make it more widely useable. The work was done on a popular computing platform so that it is easily accessible to other workers. The code is available by anonymous FTP.

For additional technical details please consult the theses of Clay Thompson and Charles Yan. Enclosed please find a copy of Charles Yan's thesis entitled "Planet Photo-Topography Using Shading and Stereo". We already submitted Clay Thompson's thesis "Robust Photo-topography by Fusing Shape-from-Shading and Stereo," last year.

Overview:

Before investing serious effort in implementation of one particular algorithm, some preliminary work on selection of the approach to recovery of topography was carried out. A variety of cost functions to be minimized were studied. Some preliminary testing was carried out on a PC using MATLAB. While a MATLAB implementation is slower than a C implementation, it is much easier to change, debug, and experiment with. Hence development time is greatly reduced.

We have developed and evaluated twelve different criteria functions. Experiments were done on synthetic shaded images, where the "ground-truth" is known accurately.

Extensive testing of all twelve different cost functions showed which are the most robust and which lead to rapid convergence. Thorough testing

showed some dependence on initial conditions and strong dependence on lighting conditions and relative orientation of the two camera positions.

A systematic comparison of a variety of cost functions showed that all of them converge on *most* of the test cases, but that the “z-only” algorithm converges fastest and has the lowest error in general. It also was able to recover shape in the case of a particularly difficult image pair that the other algorithms could not handle. These results are documented in Clay Thompson’s Ph.D. thesis.

Charles Yan then ported Clay Thompson’s Mathematica code to C. Testing on real image pairs followed. The algorithm was found to be sensitive to errors in assumed light source position as well as camera geometry; which is not too surprising. Mike Caplinger (then at Arizona State University in Tempe, AZ) helped us debug the code for calculating source and viewing directions in the transformed images that we worked with.

The results on some of the later image pairs were not as good as on the first few that we worked with. Sometimes features in some part of the surface are not recovered even though it stands out visually. In these cases we have found that while the influence of shading is quite strong, contributions from stereo information are weak—at least when far from the solution.

Charles Yan worked on resolving these issues and discovering the circumstances under which convergence can slow.

Finally we worked on applying similar techniques to other image analysis problems of interest to NASA. Stanley Brown and Gideon Stein, for example, worked on the estimation of optical flow in image sequences of Jupiter.

Analysis:

Different approaches to integrating information from shading and stereo have been explored. There were some important questions regarding the form of data representation that needed to be answered first.

Should the coordinate system be related to one of the two camera positions or “cyclopic”? Should there be independent representation for height and gradient? (In the noise-free case one should be the integral of the other). Should there be two depth-maps, or one for each camera? We found that the ‘z-only’ algorithm worked best and it requires a “cyclopic” representation.

The tests indicated clearly that the remaining key problem is one of local minima. On the synthetic data, excellent results were obtained when a reasonable starting state was used. Convergence to correct solutions was rapid. If no *a priori* information is available, convergence is not always assured. Fortunately in practice, quite a bit of information is available about the surface being viewed.

We have characterised what imaging situations make it easier to recover shape and albedo and what lighting conditions make it hard. The method works better when the lighting conditions in the two images are *different*, which is the opposite of what one finds with binocular stereo, where the usual correlation method work only if the lighting is the *same*.

We explored local shading recovery methods in an attempt to get good starting values and hence avoid local minima. We attempted to get some curvature information locally, even when height and gradient cannot be recovered locally. Simple correlation methods can then be used to get a coarse depth map by matching “curvature maps” rather than image brightness itself (Obviously correlation methods on image brightness itself will not work, because of the large difference in lighting conditions).

After careful testing of a number of competing algorithms, one was selected that is the most robust—converging most to reliability. This and other algorithms were tested on numerous synthetic image pairs which represented various difficulty levels. In the process a lot was learnt about what combinations of lighting and viewing conditions made the problem ‘ill-posed.’

After completing this investigation, the new algorithm was extended to also deal with spatially varying albedo. Some sample synthetic image pairs were generated assuming varying albedo. The extended algorithm correctly recovered both shape and albedo.

Finally the algorithm was tried on a Viking image pair provided to us by Mike Caplinger (now working on the replacement for the ill-fated Mars Observer). What appears to be a reasonable shape was covered, although quantitative comparisons are not possible, since the “ground truth” is not available in this case.

Charles Yan has ported Clay Thompson algorithm into C on the Sun workstation. He has also sought additional suitable Viking image pairs. Despite help from JPL, we have however not been able to identify which images are of the same area (Somehow better search and cataloging methods are needed for the huge data-base of space images).

We have done some further testing on the C code translation of the ‘Shape from Shading and Stereo’ algorithms. This code has now been put up on the InterNet for anonymous FTP so that other reasearches can more easily gain access to it. The FTP server is `ftp.ai.mit.edu` and the directory is `/pub/yan`. The C implementation of the algorithm is described in Xiaojian (Charles) Yan thesis entitled “Integrating Multiple Cues,” while the original MATLAB implementation is described in Clay Thompson’s thesis entitled “Shape from Shading and Stereo.” The theses are available in the same directory.

Unfortunately, the demise of the Mars Observer last August has made it impossible to continue with one of the planned stages of testing on real images. Nevertheless, Mike Caplinger is determining the utility of this algorithm for future Mars missions, should the funding requested by NASA for them be approved by Congress (The MOC spare may be flown on a 1996 launch). A more immediate interest of his is in applying this approach to proposed asteroid missions, where typically rather limited coverage is available, and there is high interest in exploiting every cue available to get accurate topographic information, even in areas where surface features needed for a purely photogrammetric approach are lacking. Shading can fill these in with relative depth information, while stereo provides the accurate absolute depth information for isolated features.

We did some preliminary work on applying the new method for integrating multiple visual cues to other domains. In the work on topography, the cues were shading and binocular stereo. Stanley Brown and Gideon Stein looked at the possibility of applying a similar approach to flow estimation on gaseous planets. While this integrated 'optical flow' algorithm works very well on synthetic data, we have the usual work to do in figuring out how to make it also work well on real data, where there are imaging artifacts, noise, changes in albedo, and other unmodelled effects.

We were hoping to get some information from Timothy Dowling that will let us determine how we can contribute to the analysis of Jupiter images in the post Shoemaker-Levy 9 era. If the features created by the impacts persist for several months and create local weather systems of some stability, then the detailed flow maps that our algorithm may be able to provide ought to be of interest.

But it appears that our funding under this contract has been terminated. We had rather hoped that the unexpended funds could be used in the fall 1994 term to finish the study of the 'optical flow' application of the method.

Future Applications:

The new method has obvious applications to the recovery of planetary topography. It combines the advantages of photoclinometry (recovery of fine detail and ability to work without need for recognizable 'features') with the advantages of binocular stereo (good absolute accuracy). Earlier work on 'Shape from Shading' has made it clear that quantitative results can be obtained — that photoclinometry is *not* just for qualitative results and 'fly through' visualization. The work here significantly strengthens the position that shading information can provide accurate information on surface topography.

Specific future applications include:

- Analysis of images from asteroid flybys, particularly where the coverage may be limited by other mission objectives.
- Analysis of images from the 'Mars Recovery Mission' (MOC2 planned for 1996).

Mike Caplinger now at Malin Space Science Systems (MSSS) in San Diego is looking into these applications already.

Theses and Reports:

- Thompson, Clay M. "Robust Photo-topography by Fusing Shape-from-Shading and Stereo," Ph.D. Thesis, MIT, Department of Mechanical Engineering. February 1993.
- Yan, XiaoJian (Charles) "Planet Photo-Topography Using Shading and Stereo," S.M. Thesis, MIT, Department of Physics, December 1993.

The source code and a 'makefile' for the 'Topography from Shading and Stereo' program may be retrieved by anonymous FTP from `ftp.ai.mit.edu` (128.52.32.11) in directory `/pub/yan`. Charles XiaoJian Yan's thesis may also be found there.

References:

Horn, B.K.P. & M.J. Brooks (1989) *Shape from Shading*, MIT Press, Cambridge, MA.

Horn, B.K.P. & M. Caplinger (1990) "Topography from Shading and Stereo," Contract Proposal.

Topography from Shading and Stereo

Berthold K.P. Horn

Artificial Intelligence Laboratory, Massachusetts Institute of Technology

Michael Caplinger

Department of Geology, Arizona State University

Abstract: *Methods exploiting photometric information in images that have been developed in machine vision can be applied to planetary imagery. Present techniques, however, focus on one visual cue, such as shading or binocular stereo, and produce results that are either not very accurate in an absolute sense or provide information only at a few points on the surface. We plan to integrate shape from shading, binocular stereo and photometric stereo to yield a robust system for recovering detailed surface shape and surface reflectance information. Such a system will be useful in producing quantitative information from the vast volume of imagery being received, as well as in helping visualize the underlying surface. The work will be carried out on a popular computing platform so that it will be easily accessible to other workers.*

1. Introduction

It would be very useful for users of image data to have automated means of extracting accurate topography from images. Existing automated methods are not able to robustly recover detailed surface shape. We propose to explore the intimate integration of existing machine vision methods to recover topography, and possibly also surface reflectance. This will be a demonstration of the application of a particular machine vision paradigm to problems of data reduction and visualization in the space sciences.

Present binocular stereo methods, whether correlation or edge-based, cannot deal with large differences in foreshortening and lighting between the two images. Conversely, so-called shape-from-shading methods cannot accurately recover the lower spatial frequency components of the topography, and may be misled by spatial variations of surface reflectance¹. The two methods are complementary in that binocular stereo can provide sparse absolute height data, while shading provides fine detail. Furthermore, binocular stereo cannot provide reliable information in areas where there is little texture, while shading works best where the surface curves smoothly and has near uniform reflectance properties.

¹What we call reflectance variations here are often referred to as albedo variations—we avoid the term “albedo,” since some people at least consider this to be a technical term with a meaning different from the one intended here.

Recent progress in shape from shading, multi-resolution grey-level matching binocular stereo, as well as photometric stereo, suggests that robust methods may be designed that combine the available information to recover accurate surface shape and reflectance information. This "fusion" of early vision modules cannot, however, be done at a point where each module has already generated its error-prone output. Instead, the available cues must be used in an integrated way. The now classic calculus of variations approach to solving early vision problems provides a means for achieving this synergism.

We will explore this new approach to integration of early vision modules in the context of the interpretation of multiple images of the same surface area obtained from different viewpoints, possibly under different lighting conditions. The result will be a system that can recover accurate, detailed topographic information, as well as surface reflectance. This will greatly enhance the value of the voluminous image data now being returned to earth from cameras on planetary explorer spacecraft, as well as earth observation platforms and from cometary flybys.

2. Applications

The volume of imaging data being received is growing all the time, and there is no hope of extracting topography from it via manual or semi-automated methods except on a piecemeal basis.

Geophysical uses of topographic data, both on Earth and other planets, typically involve the study of the movement of material over the surface under the influence of gravity. Such movement is both influenced by the existing topography and the cause of topographic changes (as in erosion). The flowing material can be water or other liquids, rock in mass movement, lava, ice, the atmosphere, or mixtures of these.

- **Hydrology:** For example, for the flow of water, the velocity of a flow that created a channel can be calculated using empirical relationships relating variables such as slope and cross-sectional area. The longitudinal profile of a stream (the slope of the fluid bed as a function of position downstream) reflects the nature of fluid processes occurring at the bed and is a sensitive measure of both surface and sub-surface sources of fluid. Such relationships have been used to explore both terrestrial water erosion and the possible genesis by water of the martian channels.
- **Mass movements:** Mass movements, such as landslides and avalanches, are caused when the gravitational force on a body of material overcomes the forces (such as shear strength) that bind it. (Sometimes, gravity is assisted by accelerations induced by earthquakes or other vibrations.)

After such movement, the volume of material moved can be computed directly from the topography, and its speed and rheological properties estimated from slope relationships and the relief of specific features on the surface of the flow.

- **Volcanism:** Lava also flows under the influence of gravity. Topographic data can be used to determine the volume of extrusion and intrusion (important in constraining processes that occur in the magma chamber), as well as to determine rheological properties of flows from the appearance of surface features. These rheological properties are in turn related to chemical and mineralogical composition, eruption rate, temperature, and other physical parameters of the lava.
- **Glaciology:** Ice, too, moves under its own weight. The slope of a ice-stream's surface reflects both the underlying topography and the rheological behavior of the ice, in particular its velocity and volume discharge rate. The relief of surface features also permits analysis of the ice depth where it cannot be directly measured.
- **Aeolian processes:** The flow of the atmosphere over topographic features is both controlled by their shape and contributes to their erosion and modification, especially if solid material is carried by the atmosphere in saltation or suspension. Knowledge of the topography can be used to estimate the nature of the flow (that is, turbulent or laminar) and the nature of material being carried by wind.

Other applications of topographic data, not related to geology directly, can also be mentioned. Some are: (a) mission planning, both on the surface and from aircraft and low-altitude satellites; (b) meteorological modeling: to constrain the surface wind field and model the generation of turbulence. Note, by the way, that much of the above applies to observations of earth also, not only to other planets.

3. Background

A considerable fraction of the bits returned to earth by spacecraft sent to explore the planets comes in the form of images. Presently the means to obtain quantitative information from these images are largely restricted to photogrammetric methods and profile-based photoclinometric methods. Yet a human observer gets a wealth of additional qualitative information from these images that suggest that it ought to be possible to extract more, using advanced machine vision techniques. If detailed surface shape and reflectance information can be recovered, it can be presented in a variety of ways to aid in visualization using well-known rendering methods from computer graphics.

We want to exploit this opportunity by applying the latest methods in work on “early vision.” First we review traditional methods of image analysis.

3.1 Photogrammetry

Photogrammetric methods permit the accurate determination of the positions of isolated points in three-dimensional space from (typically manual) spot measurements of corresponding images of well defined features on the surface. This is a very important basic step in defining the surface, and in recovering the camera position and attitude, but provides information at only a small number of isolated points, which do not define the detailed shape in between. Nevertheless, sound photogrammetric techniques are vital to the determination of an accurate reference body for use in cartography and in determining the relative positions and orientations of exposure stations used to obtain image pairs for binocular stereo [Horn 89].

3.2 Photoclinometry

Present photoclinometric methods permit the recovery of isolated profiles across features that have special symmetries, such as circular craters, volcanic calderas, and linear depressions or grabens [Bonner & Schmall 73] [Davis *et al.* 82] [Davis & Soderblom 83] [Davis & McEwen 84] [Howard *et al.* 82] [Lambiotte & Taylor 67] [Lucchitta & Gambell 70] [Malin & Danielson 84] [McEwen 85] [Passey & Shoemaker 82] [Rowan *et al.* 71] [Tyler *et al.* 71] [Watson 68] [Wildey 86] [Wilhelms 64] [Wilson *et al.* 84] (for a larger collection of references on this subject, see [Horn & Brooks 90]). Since these methods cannot take into account cross-profile inclination, they will not produce accurate profiles if the cross-profile inclination is non-zero. Hence such methods are limited to areas that have the appropriate symmetry. Shape-from-shading, basic to what is being proposed here, may be thought of as “area-based” photoclinometry. It permits the recovery of complex, wrinkled surface shapes by using image information from a full two-dimensional region of the image, rather than merely along a line in the image. There is no restriction to surfaces having predefined symmetries (see section 4.2).

3.3 Correlation-based Stereo

There have been many attempts to automate the recovery of topographic information from two images using binocular stereo methods. Perhaps the

oldest and most widely known are methods based on correlation. A wide-variety of techniques, ranging from Fourier transforms to optical computing have been employed to speed the computation [Horn 86]. Such methods assume that, locally at least, what appears in one image is a shifted version of what appears in the other image. If, however, the surface is viewed from quite different viewpoints, the foreshortening in the two images will be very different and these methods will fail, since they are not designed to match two waveforms that have been stretched by different amounts [Horn 83]. Such methods are thus restricted to situations where the baseline-to-height ratio is small. Unfortunately, the expected error in the determination of the distance from the camera to the surface grows inversely as the baseline-to-height ratio—so these methods have not proven useful in the interpretation of aerial photographs, for example, where a large baseline-to-height ratio is purposefully employed to get accuracy in the determination of height comparable to the accuracy in the determination of the horizontal position. Furthermore, in planetary exploration it is common to have two images of the same area taken not only from very different viewpoints, but under very different lighting conditions. Correlation-based binocular stereo is based on the assumption that the patterns of brightness in the two images are the same and so cannot be applied successfully in this situation.

Of the dozens of attempts over the past twenty years to automate binocular stereo using this kind of approach, apparently the only one that comes close to being useful is embodied in the so-called Gestalt photomapper. On undulating terrain with sufficient texture and no confusing reflections from specular surfaces such as lakes, this machine can produce beautifully detailed surface topography. It does, however, require considerable assistance from an experienced operator in order to help it out in difficult areas. It also does not work satisfactorily in areas of steep relief or where there is insufficient contrast in texture patterns. While the detailed inner workings of the machine are proprietary, it is known that part of the reason that it works at all is that it is not based on blind correlation, but an iterative scheme that warps the local surface in a hexagonal patch using a high-order polynomial. In this fashion it can take account of the differences in foreshortening, provided that the initial guess is close enough to the correct shape to allow it to converge.

3.4 Edge-Based Stereo

To overcome the problems of unequal foreshortening and unequal brightness, there has been considerable work on feature-based binocular stereo in the last ten years. In this case distinctive “features” of the images (such as rapid tran-

sitions in grey level) are first extracted and then these features are matched symbolically. The matching process is complex and requires representation of the images at multiple resolutions. It does not lend itself readily to parallel implementation.

The basic assumption in feature-based binocular stereo is that image features correspond to distinguished surface features (such as terrain breaks) and that their image position is not affected by vagaries of lighting and foreshortening. With low baseline-to-height ratio and similar lighting in the two images, such methods can produce reasonable estimates of height along isolated curves. The remaining surface is unknown and has to be somehow interpolated from the sparse data recovered [Grimson 82]. This means that in areas where there are few “features” very little is really known about local surface topography.

4. Photometric Machine Vision Methods

In machine vision, a number of methods have been developed that exploit the fact that a grey-level is a quantized estimate of image irradiance, and that image irradiance depends on surface orientation, surface reflectance and light source distribution. Such methods have shown promise in applications to aerial photographs, satellite photographs of the earth and planetary images. We briefly review the relevant approaches here, noting that what we are proposing now is an integration of these methods—but not one that merely builds a system out of several existing modules. We are not proposing to merely combine the outputs of various modules in some simple statistical way. Instead, the calculus of variations approach is to be used to minimize some overall error function. We believe that this is the way to obtain the necessary synergism between different visual cues.

4.1 Photometric Stereo

Images of an object taken from the same viewpoint but under different lighting conditions can be used to recover surface shape. This is called *photometric stereo*, in distinction to *binocular stereo* discussed above [Horn *et al.* 78] [Horn 86]. The basic idea is that local surface orientation on a more or less smooth surface can be specified using two parameters (such as the slope in the x -direction and the slope in the y -direction). A measurement of brightness at the corresponding image point places a single constraint on the surface orientation—not enough to recover it fully. To recover surface orientation, various other sources of additional constraint can be exploited. Perhaps the

easiest is the brightness, at the same point in an image, taken when the object is illuminated differently.

With two images, two constraints are available—that is, it becomes possible to locally recover the surface orientation. The computation can even be implemented in a lookup table indexed on the two grey-levels at the same picture cell. The lookup table can be constructed using some theoretical photometric model and known illumination conditions, or it can be developed by numerical inverting calibration data obtained from an object of known shape.

One can do even better if more than two images are available. If, for example, the surface reflectance is spatially varying, then two images are not sufficient to locally recover the three degrees of freedom, but three images will do. Continuing in this way, if the photometric model has n unknown parameters that may vary from point to point, then $(n + 2)$ images are needed to estimate these parameters, as well as the two components of surface orientation².

4.2 Shape from Shading

Shape from shading is the recovery of surface shape from shading in a single image. Shading is the spatial variation of image brightness resulting from corresponding variations in surface orientations. The twenty year history of work in this area is captured in a recently published collection of papers called *Shape from Shading* [Horn & Brooks 89]. This book also contains a complete bibliography of the three hundred or so publications in this, and related fields, such as photoclinometry—so we will keep the list of references short here. For a quick introduction to the subject, see chapters 10 & 11 in *Robot Vision* [Horn 86].

In shape from shading the local ambiguity in surface orientation, described above, is resolved by assuming that the surface “hangs together,” that is, neighboring surface patches are not allowed to have totally unrelated orientations. Put another way, if one were to walk in a closed path atop the surface patches one ought to come back to the same height. Use of this additional constraint is, however, much more difficult than use of information from a second image as in photometric stereo. It has taken a long time for theory and implementation to mature to where these methods are really practical on other than synthetic images and real images of relatively simple shapes³.

²There are limitations to this in the context of planetary imaging, since the light source, at least over a short period of time, moves along a small circle on the sphere of directions, and thus does not sample the space of all possible source directions well.

³Even at this stage, question of existence and uniqueness of the solution are re-

Nevertheless, robust methods that work on real images of complex, wrinkled surface have been developed in recent years [Frankot & Chellapa 88] [Szeliski 90] [Zheng & Chellapa 90]. An example of what can be done is the surface recovered from a SPOT image of a hilly area east of Huntsville, Alabama by the new method in [Horn 90] (See section 8 and figures attached at the end). A reasonable surface shape was recovered despite the fact that the range of distinct grey-levels in the region of interest was only 19, that the photometric function, the light-source position and the atmospheric state were not accurately known, and that the image was noisy and appeared to suffer from the effects of aliasing. The recovered shape did not reproduce the lower spatial frequencies of the terrain very accurately, as expected. Nevertheless, a contour map obtained from a smoothed version of the surface looks similar to the appropriate part of the USGS topographic map of the same area. Conversely, fine surface undulations were recovered that do now show up in the “generalized” topographic map.

Perhaps the earliest work in this field can be traced back to [Rindfleisch 66], who found that very specialized assumptions about the surface reflectance properties (namely that brightness is a linear combination of the slope in the x -direction and the slope in the y -direction) allows one to estimate a profile of the surface by simple integration. Some recent work also uses this assumption, which dramatically simplifies the problem [Pentland 88]. But this is not a realistic assumption, nor is it desirable [Horn 70]. The reason is that the recovered profiles are not connected to one another, each can independently float up or down with respect to its neighbor, so that there is tremendous ambiguity in the recovered surface.

If surface slopes are small, then reflectance properties may be locally linearized and the above used as an approximation. It has been found that more general purpose shape-from-shading methods work particularly well when the slope excursion are small [Kirk 84, 87].

Shape-from-shading suffers from some of the same limitation as photoclinometry, since it is also based on an assumed known photometric function. However, since information from a two-dimensional region is used in a least-squares approach, the effects of image noise and errors in photometry tend to be suppressed. Conversely, photoclinometry is limited because varying surface reflectance along a profile cannot be distinguished from changes in brightness due to surface orientation. When a two-dimensional patch is considered, on the other hand, such variations lead to inconsistencies that can be detected and at least used to flag the area as suspect. Hopefully we will learn more

ceiving considerable attention—as are issues of what kind of boundary conditions might be needed. These do not concern us very much here, since we know that there is a solution, and we can often get a good first estimate of it.

about this in future to actually be able to recover both shape and surface reflectance—although at this stage this does not look too promising since the problem appears to be underconstrained when working from a single image.

4.3 Grey-level Matching Stereo

People have little difficulty interpreting stereo pairs of smooth surfaces lacking the obvious “features” needed by present feature-based binocular stereo methods. As a result, there has been considerable interest in new methods that have been developed to directly match grey-levels, or information derived directly from grey-levels using local operators [Gennert 86] [Barnard 89]. Such methods hold the promise of providing dense surface information, although they have to overcome the problems inherent in the assumption that a given point in the scene will yield similar grey-levels in the two images. It is known that such methods cannot work on a single level of resolution or they will immediately get locked into a local minimum resulting from inappropriate matches between the two images. Multi-resolution techniques are needed to solve this problem.

5. Proposed New Method

There has been quite a bit of work recently on sensor “fusion” and early vision module “integration,” driven by the lack of robustness of individual modules. However, most of this work either uses the results of one module to constraint the operation of another, or simply combines the results of several module using some simple least-squares weighting function. Neither of these approaches has proven particularly useful. We propose instead to integrate the modules much more intimately. Over the past ten years, a new approach to machine vision has arisen, based on the calculus of variations, as pioneered in [Horn & Schunck 81] and [Ikeuchi & Horn 81]. Typically an error functional is defined that has several component penalty functions that allow one to express a preference for solutions having certain properties, such as smoothness. This helps one deal with what would otherwise be underconstrained or ill-conditioned problems.

What we propose to do is to build an energy functional that contains penalty terms for errors in shading in the left image, errors in shading in the right image and errors in matching left and right image. We will then have to find the equations governing the resulting variational problem and develop methods for iteratively obtaining a solution to a discrete version of the resulting partial differential equations. The hope is that the problem can

be formulated in such a way that the not inconsiderable investment in work on the latest shape-from-shading system [Horn 90] can be somehow carried over into the new system for shape from shading and binocular stereo.

Initially the main concern will not be computational efficiency, except in so far as to avoid methods that will not later lend themselves to speedup using numerical approaches such as multigrid and gradient descent methods, or massively parallel hardware such as the Connection Machine. On a low-end work-station we do not expect to be able to process large images in a short time. The focus will be on developing a sound mathematical approach first. We do plan, however, to incorporate, as soon as possible, ideas on high speed implementations of the iterative algorithms being pursued by [Szeliski 90] [Zheng & Chellapa 90].

The resulting system will be tested both on synthetic image pairs and real image pairs. The reason for working with synthetic image data first is that this is the only situation in which “ground truth” is known absolutely. But of course a system that only works on clean, noise-free synthetic data is of little interest, so test will have to be made on real data also. We propose to do this first with satellite images of the surface of the earth, again because the “ground truth” is more readily accessible to us in this case. Finally, the system will be tested on existing planetary images.

5.1 Limitations on Accuracy

As mentioned before, shape-from-shading methods can provide fine detail, but not accurately reproduce the lower spatial frequency undulations of the terrain. Conversely, binocular stereo can give good spot absolute height measurements, but cannot provide dense coverage of the surface, particularly in areas without sufficient texture. We expect that the integration of these two modalities will overcome the short-comings of both method. There will, however, still be limitations resulting from short-comings of the imaging systems and limited knowledge of surface photometry.

In profile-based photoclinometry, the effects of errors in photometry are severe, since the measured brightness is directly translated into surface slope. There is no way locally to tell the difference between a change of brightness resulting from a spatial variation in surface reflectance and one resulting from a change in surface orientation. In area-based shape-from-shading, information from a two-dimensional image region is used, and certain errors tend to cancel out. At the very least, the inability to reduce the error functional flags the solution as suspect. Furthermore, when this method is integrated with binocular stereo, it can be expected that the lower spatial frequencies in the

reconstruction will be largely controlled by the stereo data and so the effects of errors in photometry will be greatly reduced.

Most of the imagery we propose to work with was acquired using vidicon-based cameras and so may suffer from poor radiometric calibration, non-linear response, spatially non-uniform response and geometric warping. Even though attempts are made to remove these effects through careful pre-processing, we still expect that the results obtained from these images will be inferior to those possible in future with CCD-based cameras.

6. Proposed Work

The proposed work can be divided into six components:

6.1 Application of Photometric Stereo

To become more familiar with photometric properties of planetary surfaces, and to explore the potential for simultaneous analysis of multiple images of the same surface, we propose to start with photometric stereo analysis on images taken with different lighting conditions but from essentially the same exposure station. It will be best at first to work with images obtained from a totally static platform. For this reason we would like to start with certain Viking lander images. We will first assume a photometric function (such as some popular combination of Hapke, Minneart, Lommel-Seeliger, and Lambertian models) and try and recover surface orientation and surface reflectance from three images. We will then work backwards from a calibration object with approximately known surface shape to a photometric function, possibly using many more than three images. Once the lookup table has been constructed it can be used to interpret the rest of the scene. Numerical photometric information obtained, using the same sensor as that used later in measurements of shape, is likely to be much more useful than that provided by an arbitrary analytical form—except in so far as that existing analytical forms may be helpful in bridging gaps in measurements and smoothing out noise.

We realize that the photometric properties of surfaces at different scales are typically very different, so that the detailed results on close range imagery will not apply directly to images taken from orbital distances. Yet the principle involved is the same, and having images taken from *exactly* the same point enables photometric stereo analysis.

6.2 Photometric Stereo and Binocular Stereo

We plan next to investigate the integration of photometric stereo (same position—different lighting) with binocular stereo (same lighting—different position). This promises to be easier to achieve than our ultimate goal of integrating shading and binocular stereo. At the same time we expect to learn valuable lessons from this exercise since some of the same mathematical tools and programming techniques come into play. We expect that even here we will need to develop suitable multi-scale algorithms to overcome the problems of local minima resulting from stereo mismatches.

It should be pointed out that integration of photometric stereo and binocular stereo on close-range image sets may have applications to autonomous vehicle control, since shapes of the surfaces in the environment can be recovered, as well as their surface reflectance patterns. It is quite likely that surface recovery using such integrated methods will involve less complex computations than those from binocular stereo alone⁴.

6.3 Application of Shape from Shading

We then propose to apply the latest shape-from-shading method to recovery of shape from existing imagery such as certain Viking Orbiter images. We plan also to work on the more complex problem of “whole disk” shape from shading using images of Deimos. This is more difficult because the boundary constraints on slope are harder to integrate—since slope becomes infinite on the occluding boundary.

6.4 Shading and Stereo—Similar Lighting, Similar Viewpoint

Next, we come to the heart of the work proposed here, the integration of shape from shading and grey-level matching stereo. We propose to first work on satellite images of the earth, since we have access to reasonably accurate ground truth in this case. We also will initially select situations where the view points are not too wildly different, surface cover is fairly uniform and the lighting is similar in the two images (for example, using a stereo pair of SPOT

⁴On the other hand, for a rapidly moving platform, integration of direct motion vision and binocular stereo is ultimately likely to be more useful still. We do not propose here to work on this, although we expect that a similar approach can be taken to integration of direct motion vision methods and grey-level matching stereo.

images of a hilly tree-covered region east of Huntsville, Alabama)⁵. The main part of the proposed work will be the development of the mathematical model and the computer implementation of the algorithm to solve this problem.

6.5 Shading and Stereo—Different Lighting, Different Viewpoint

Finally, we propose to extend the integrated shading and binocular stereo approach to harder cases, where surface reflectance may vary, viewpoints may be widely spaced and illumination very different in the images. This in essence will require an integration of photometric stereo methods with shape from shading and binocular stereo. We hope that the approach can be extended to certain Voyager images where the photometric situation is more complex than on the rocky planets of the inner solar system.

6.6 More than two images

Once we have formalized the techniques for integrating shape from shading and binocular stereo in two images, we will consider the solution of problems where more than two images taken from different viewpoints under different lighting conditions are available. We expect that the additional constraint provided will make solutions even more robust, provided one can develop a procedure to get close enough to the solution so convergence is guaranteed. In this situation integration of methods from all three areas: shape from shading, photometric stereo and binocular stereo will be appropriate.

6.7 High Speed Implementation

We expect initially to work with relatively small images or parts of images in order to allow debugging with a reasonable turn around time on a small workstation. Until we have solved the underlying mathematical and numerical problems, and demonstrated reasonable algorithms, we will not place great emphasis on implementations that buy speed in return for complexity. Finally, however, for the results of this work to be useful on high resolution images, attention will have to be directed to methods for reducing the number of iterations required, such as multigrid and gradient descent approaches [Szeliski 90] [Zheng & Chellapa 90].

⁵We may also try to work with NOAA polar orbiter images here, since they provide stereo coverage at considerably lower cost, but expect that the extremely low resolution side-lap stereo coverage will lead to quite unimpressive results.

7. Requirement and Deliverables

To do what we proposed to do above, we will need access to the indicated image sources. To summarize:

- Viking Lander images (for the initial photometric stereo work).
- Viking Orbiter images of the Martian surface, first for application of the existing shape-from-shading algorithm, and later for further testing of the new integrated algorithm.
- Viking Orbiter images of Martian satellites for whole-disk shape from shading.
- SPOT earth images for usable quality integration of shading and stereo.
- Voyager images for work on images with more difficult photometry.

7.1 Hardware and Software

For this work to have a impact on the planetary science community it will have to be available on a widely-used platform—we cannot expect others to re-implement what is going to be a non-trivial software system. Presently the latest shape-from-shading work is implemented in Common Lisp on a Symbolics Lisp Machine. This machine has a software environment very conducive to rapid prototyping and debugging of software, but it is not widely available. We propose to do all future work in the programming language C on a SPARC station from Sun Microsystems. To this end, we propose to acquire the following hardware:

- Graphics SPARC station from Sun Microsystems.
- CDROM drive to read the proposed Voyager image library.
- DAT drive to read Viking Orbiter data and store other image libraries.
- Scanner for images that are not available in machine readable form.

We have file servers, laser printers, standard 9-track tape drives and software in place to support the work we plan to do. We also have a high quality stereo viewer (although no stereo-comparator or other means of accurate image measurement). We do not have a means of high quality hard-copy graphics output, but expect that for demonstration purposes half-tone output on well-tuned laser printers will be adequate.

We will have to acquire at least one SPOT image stereo pair for this work. Travel to one conference per year should be covered so we can present the results of the work. We will also need to cover travel for one trip from MIT to ASU, and one trip from ASU to MIT per year to facilitate detailed collaboration on this project.

7.2 Technology Transfer

By working on a popular platform, using a widely known programming language, we provide for ease of transfer of what we develop to other sites. The photometric stereo program, the new shape-from-shading program and the integrated shading and binocular stereo system will all be made available. If the integrated photometric stereo and binocular stereo system turns out to work well and be useful (other than as a stepping stone), it too will be made available. This proposal does not, however, request resources that would be needed to provide software support beyond documentation supplied with the programs and technical papers describing the algorithms.

8. Illustrations of Previous Related Work

We attach two representative examples of earlier work on the application of shape-from-shading methods to images of interest to planetary and earth scientists.

Shown in Figure 1a are two images of Deimos taken by Viking Orbiter (339B02 & 428B22). In Figure 1b are shown corresponding surface reconstructions using an implementation by Michael Caplinger of an older shape-from-shading algorithm [Horn & Brooks 85]. Brighter areas are closer to the viewer than darker ones.

Shown in Figure 2a is a stereo pair of portions of two satellite images of Monte Sano State park east of Hunstville, Alabama. In Figure 2b is a synthetic stereo pair constructed using the topography recovered by a shape-from-shading algorithm developed recently by Berthold K.P. Horn [Horn 90]. The input to the algorithm was the left sub-image of Figure 2a (the right sub-image was *not* used). Figure 3a is a portion of the USGS topographic map covering the area, while Figure 3b is a contour map created from a smoothed version of the digital **terrain** model recovered using shape from shading.

These examples, while based purely on shape from shading, at once illustrate the promise of such approaches and also show some of the short-comings discussed earlier, which we plan to overcome by intimately integrating binocular stereo with shape from shading.

9. Summary

We plan to intimately integrate shape from shading and grey-level matching stereo to obtain robust recovery of surface topography and surface reflectance from multiple images of planetary targets. In preparation for this, we will (a)

demonstrate photometric stereo applied to images obtained from the same view point but under different lighting conditions, (b) demonstrate the latest shape-from-shading algorithm on real planetary images, (c) demonstrate the potential for recovering surface topography and surface reflectance from satellite images of the earth and other planets.

The main contribution of this work will be a considerable increase in the value of existing and planned planetary imagery resulting from the ability to get important quantitative information in an automated fashion. Amongst other things, this will enable quantitative analysis and better means of visualizing the data.

10. References

- Barnard, S. (1989) "Stochastic Stereo Matching over Scale," *International Journal of Computer Vision*, Vol. 3, pp. 17-32.
- Bonner, W.J. & R.A. Schmall (1973) "A Photometric Technique for Determining Planetary Slopes from Orbital Photographs," U.S. Geological Survey Professional Paper 812-A, pp. 1-16.
- Davis, P.A., L.A. Soderblom & E.M. Eliason (1982) "Rapid Estimation of Martian Topography from Viking Orbiter Image Photometry," in *Reports on Planetary Geology Program* (1982), NASA Technical Memorandum 85127, pp. 331-332.
- Davis, P.A. & L.A. Soderblom (1983) "Rapid Extraction of Relative Topography from Viking Orbiter Images: II. Application to Irregular Topographic Features," in *Reports on Planetary Geology Program* (1983), NASA Technical Memorandum 86246, pp. 29-30.
- Davis, P.A. & A.S. McEwen (1984) "Photoclinometry: Analysis of Inherent Errors and Implications for Topographic Measurement," *Lunar and Planetary Science Conference XV*, 12-16 March, pp. 194-195.
- Frankot, R.T. & R. Chellapa (1988) "A Method for Enforcing Integrability in Shape from Shading Algorithms," *IEEE Transactions on Pattern Analysis and Machine Intelligence*. Vol. 10, No. 4, pp. 439-451, July. Also *Shape from Shading*, Horn, B.K.P. & M.J. Brooks (eds.), MIT Press, Cambridge, MA..
- Gennert, M. A. (1987) "A Computational Framework for Understanding Problems in Stereo Vision," Ph.D. Thesis, Department of Electrical Engineering and Computer Science, MIT, Cambridge, MA.
- Grimson, W.E.L. (1982) "A Computational Theory of Visual Surface Interpolation," *Philosophical Transactions of the Royal Society of London B*, Vol. 298, pp. 395-427.

- Horn, B.K.P. (1970) "Shape from Shading: a Method for Obtaining the Shape of a Smooth Opaque Object from One View," Technical Report TR-79, Project MAC, MIT, Cambridge, MA. Technical Report TR-232, MIT AI Laboratory, Cambridge, MA. Also in *Shape from Shading*, Horn, B.K.P. & M.J. Brooks (eds.), MIT Press, Cambridge, MA..
- Horn, B.K.P. (1983) "Non-Correlation Methods for Stereo Matching," *Photogrammetric Engineering and Remote Sensing*, Vol. 49, No. 4, pp. 535-536, April.
- Horn, B.K.P. (1986) *Robot Vision* (particularly Chapters 10 & 11, pp. 202-277), MIT Press, Cambridge, MA & McGraw-Hill, New York, NY.
- Horn, B.K.P. (1989) "Relative Orientation," to appear in *International Journal of Computer Vision*, Also (1987) Memo 994, MIT AI Laboratory, Cambridge, MA.
- Horn, B.K.P. (1990) "Height and Gradient from Shading," to appear in *International Journal of Computer Vision*. Also, (1989) Memo 1105, MIT AI Laboratory, Cambridge, MA.
- Horn, B.K.P. & M.J. Brooks (1986) "The Variational Approach to Shape from Shading," *Computer Vision, Graphics and Image Processing*, Vol. 33, No. 2, pp. 174-208, February. Also *Shape from Shading*, Horn, B.K.P. & M.J. Brooks (eds.), MIT Press, Cambridge, MA..
- Horn, B.K.P. & M.J. Brooks (1989) *Shape from Shading*, MIT Press, Cambridge, MA.
- Horn, B.K.P. & B.G. Schunck (1981) "Determining Optical Flow," *Artificial Intelligence*, Vol. 17, No. 1-3, August 1981, pp. 185-203. Also (1980) Memo 572, MIT AI Laboratory, Cambridge, MA. Also AD-A093925 available from NTIS.
- Horn, B.K.P., R.J. Woodham & W.M. Silver (1978) "Determining Shape and Reflectance using Multiple Images," Memo 490, MIT AI Laboratory, Cambridge, MA. Also AD-A062380 available from NTIS.
- Howard, A.D., K.R. Blasius, & J.A. Cutt (1982) "Photoclinometric Determination of the Topography of the Martian North Polar Cap," *Icarus*, Vol. 50, pp. 245-258.
- Ikeuchi, K. & B.K.P. Horn (1981) "Numerical Shape from Shading and Occluding Boundaries," *Artificial Intelligence*, Vol. 17, No. 1-3, pp. 141-184, August. Also in *Computer Vision*, Brady, J.M. (ed.), North Holland. Also in *Shape from Shading*, Horn, B.K.P. & M.J. Brooks (eds.), MIT Press, Cambridge, MA..
- Kirk, R.L. (1984) "A Fast Finite-Element Algorithm for Two-Dimensional Photoclinometry," *Bulletin of the American Astronomical Society*, Vol. 16, No. 3, pg. 709.
- Kirk, R.L. (1987) "A Fast Finite-Element Algorithm for Two-Dimensional Photoclinometry," Part III of Ph.D. Thesis, Division of Geological and Planetary Sciences, California Institute of Technology, Pasadena, CA.
- Lambiotte, J.J. & G.R. Taylor (1967) "A Photometric Technique for Deriving Slopes from Lunar Orbiter Photography," *Proceedings of the Conference on the Use of Space Systems for Planetary Geology and Geophysics*, Boston, MA, May 25-27.

- Lucchitta, B.K. & N.A. Gambell (1970) "Evaluation of Photoclinometric Profile Determination," in *Analysis of Apollo 8 Photographs and Visual Observations*, NASA SP-201, National Aeronautics and Space Administration, pp. 51-59.
- Malin, M.C. & G.E. Danielson (1984) "Topography on Ganymede Derived from Photoclinometry," in *Reports on Planetary Geology Program* (1983), NASA Technical Memorandum 86246, pp. 29-30.
- McEwen, A.S. (1985) "Topography and Albedo of Ius Chasma, Mars," *Lunar and Planetary Science Conference XVI*, 11-15 March, pp. 528-529.
- Passey, Q.R. & E.M. Shoemaker (1982) "Craters and Basins on Ganymede and Callisto: Morphological Indicators of Crustal Evolution," in Morrison, D. (ed.) *Satellites of Jupiter*, University of Arizona Press, Tucson, pp. 379-434.
- Pentland, A.P. (1988) "Shape Information from Shading: A Theory about Human Perception," Technical Report 103, Vision Sciences, MIT Media Laboratory, MIT, Cambridge, MA, May.
- Rindfleisch, T. (1966) "Photometric Method for Lunar Topography," *Photogrammetric Engineering*, Vol. 32, No. 2, pp. 262-277, March.
- Rowan, L.C., J.F. McCauley & E.A. Holm (1971) "Lunar Terrain Mapping and Relative Roughness Analysis," U.S. Geological Survey Professional Paper 599-G, pp. 1-32.
- Tyler, G.L., R.A. Simpson & H.J. Moore (1971) "Lunar Slope Distributions: Comparison of Bi-Static Radar and Photographic Results," *Journal of Geophysical Research*, Vol. 76, No. 11, pp. 2790-2795.
- Szeliski, R. (1990) "Fast Shape from Shading," Internal Report, DEC Cambridge Research Center, Cambridge, MA.
- Watson, K. (1968) "Photoclinometry from Spacecraft Images," U.S. Geological Survey Professional Paper 599-B, pp. 1-10.
- Wildey, R.L. (1986) "Radarclinometry for the Venus Radar Mapper," *Photogrammetric Engineering and Remote Sensing*, Vol. 52, No. 1, pp. 41-50, January. Also in *Shape from Shading*, Horn, B.K.P. & M.J. Brooks (eds.), MIT Press, Cambridge, MA..
- Wilhelms, D.E. (1964) "A Photometric Technique for Measurement of Lunar Slopes," in *Astrogeological Studies Annual Progress Report, Part D: Studies for Space Flight Program*. U.S. Geological Survey Open-File Report, pp. 1-12, May, NASA Catalog Number N66 35597.
- Wilson, L., M.A. Brown, E.M. Parmentier & J.W. Head (1984) "Theoretical Aspects of Photoclinometry Terrain Profiling on the Galilean Satellites," in *Reports on Planetary Geology Program* (1983), NASA Technical Memorandum 86246, pp. 27-28.
- Zheng Q. & R. Chellapa (1990) "A Robust Algorithm for Inferring Shape from a Single Image," Working Paper, USC, SIPI, Los Angeles, CA.

11. Timetable and Milestones

11.1 First Year

- Code photometric stereo method and numerical derivation of lookup table from “calibration object.” Apply to Viking Lander Images. Determine photometry and shape. Compare with existing depth maps obtained manually using binocular stereo.
- Formulate problem of integrating photometric stereo with (grey-level matching) binocular stereo. Experiment with proposed approaches on Viking Lander images.
- Recode the latest shape from shading algorithm [Horn 90] for the workstation. Apply to Viking Orbiter images of surface features and to whole-disk images of Martian satellites.

11.2 Second Year

- Solve problem of integrating photometric stereo with (grey-level matching) binocular stereo. Find robust solution method. Experiment with solutions on Viking Lander images.
- Formulate problem of integrating shape from shading and binocular stereo for similar lighting and similar viewpoint case. Solve resulting variational problem. Develop discrete approximation and iterative scheme for solving same. Experiment with SPOT satellite images.
- Develop methods for detecting areas with spatial variations in reflectance.

11.3 Third Year

- Formulate problem of integrating shape from shading and binocular stereo for different lighting and vastly differing viewpoint case. Solve resulting variational problem. Develop discrete approximation and iterative scheme for solving same. Experiment with Viking Orbiter images.
- Integrate shape from shading, photometric stereo and binocular stereo to increase robustness when spatial variations in reflectance are large.
- Extend the integrated method to deal with more than two images. Extend to situations where the photometry is more complex (icy surfaces instead of rocky surfaces).

12. Hardware Choice Justification

We need to implement the algorithms coming out of the research and development proposed here on some widely available platform, having a C compiler and running Unix. The Sun SPARC Station has adequate power and is inexpensive, being the cheapest in their line of work stations. The package containing some local storage (105 Mbyte disk) will be required in order to reduce network traffic in accessing images being worked with, as well as image-registered data such as digital terrain models, gradient images and surface reflectance maps.

We wish to use the version of the workstation that has a 19" monitor and grey-level capability, since we will be displaying images and intermediate results of the computations frequently (Dithering of images on a binary display screen is slow and does not yield results that permit assessment of the quality of the computation). Finally, the graphics version of this low-end system has somewhat higher speed for the common operations used in visualization of the data in comparison with the rather slow rate for such operations on the basic color SPARC station.

Working with multiple images and registered arrays of height, surface orientation, reflectance and disparity information, will be greatly aided by having 16 Mbytes of memory, given that a some part of the memory will be taken up by the operating system and the complex software environment we will be constructing.

To read some of the image sources (such as the Voyager image library) we need a CDROM drive (which is read only, of course). To read other images, and to permit exchange of images between the two principal investigators and others, we also need a 4mm DAT drive (which is read/write).

Finally, there are situations where it is not feasible to obtain an image in machine readable form, yet a high quality photographic print is available. To deal with this situation we would like to use a full grey-level flat-bed scanner.

Planet Photo-Topography Using Shading and Stereo

by

Charles XiaoJian Yan

B.S., California Institute of Technology, 1990

Submitted to the Department of Physics
in partial fulfillment of the requirements for the degree of

Master of Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

December 1993

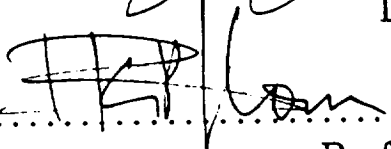
© 1993 Charles XiaoJian Yan. All rights reserved.

The author hereby grants to MIT permission to reproduce and to distribute copies of this thesis document in whole or in part.

Signature of Author 

Department of Physics

December 15, 1993

Certified by 

Prof. Berthold K. P. Horn

Professor of Electrical Engineering and Computer Science

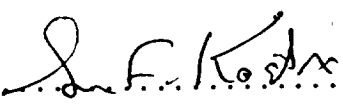
Thesis Supervisor

Certified by 

Prof. Robert P. Redwine

Professor of Physics

Thesis Supervisor

Accepted by 

Prof. G. F. Koster

Chairman, Departmental Committee on Graduate Students

This page is left blank intently.

Planet Photo-Topography Using Shading and Stereo

by

Charles XiaoJian Yan

Submitted to the Department of Physics
on December 15, 1993, in partial fulfillment of the
requirements for the degree of
Master of Science

Abstract

Two of the most successful methods in Computer Vision developed over the years are Shading and Stereo. Newly proposed DFSS (Depth From Shading and Stereo) method for fusing the two is examined. The most robust z-only method is utilized and explored in analyzing planetary data images, namely the Mars image data from Viking Space Project. Two major extensions to DFSS are to integrate all the image cells, instead of single test cell to increase robustness, and to use realistic Mars images, instead of synthetic test images. Another unique feature presented in this thesis is in dealing with real world images, containing noise, geometry error, calibration errors, and reflectance errors.

Thesis Supervisor: Prof. Berthold K. P. Horn

Title: Professor of Electrical Engineering and Computer Science

Thesis Supervisor: Prof. Robert P. Redwine

Title: Professor of Physics

This page is left blank intently.

Acknowledgments

First and foremost, I would like to thank my thesis advisor, Professor Berthold K. P. Horn, for his invaluable guidance, support, understanding and inspiration. Aside from academic and research help, Professor Horn is a very kind and gentle person. I am also thankful to Professor Robert Redwine, who is not only research supervisor but also academic advisor. Professor Redwine has listen to the problems I encountered during the years, helped to solve them, steered me towards a mature professional.

Also, I am grateful to Dr. Mike Caplinger. Dr. Caplinger has put in much time to select and process the images pair for this research. He has given many helpful suggestions during this research. Without Dr. Caplinger's help, the project will never succeed.

Also thanks go to fellow students at the AI lab for the discussions and suggestions, especially Dr. Clay Thompson.

Last, certainly not the least, I feel deeply in debt to the support, understanding, and help from my wife, Ellen Hui Ding, my parents, Professor WuGuang Yan and Professor WenXiu Gao. Without their continuous caring and encouragement, I can never be wherever I am today.

This research was supported in part by NASA contract NAS5-31352. Additional support came from DARPA contract N00014-91-J-4038 which helps support general activities at the M.I.T. Artificial Intelligence Laboratory.

This page is left blank intently.

Contents

1	Introduction	19
1.1	Background	20
1.2	Planet Photo-Topography	23
1.3	Viking Project, Planet Image Data	25
1.3.1	Viking Mission	25
1.3.2	Viking Orbiter Visual Subsystem	25
1.4	Related Research	26
1.5	About This Thesis	26
2	Overview	28
2.1	Coordinate System	28
2.1.1	Perspective Projection	29
2.1.2	Othographic Projection	31
2.2	Shape From Shading	32
2.2.1	Image Formation Process	32
2.2.2	Object radiance stage	33
2.2.3	Image Generation Stage	36
2.3	Stereo	36
2.4	Photo-Topography	39
2.5	Simplification	41
2.5.1	Special Global Coordinate System	41

<i>CONTENTS</i>	8
2.5.2 Removing the View Direction Dependence	42
2.5.3 Constant Albedo	43
2.5.4 Aligned Cameras	44
2.5.5 Summary of Simplified equations	45
2.6 Camera Calibration	47
3 Fusion of Shape from Shading and Stereo	48
3.1 Fusion Strategy	50
3.2 <i>z-only</i> DFSS Algorithm	51
3.2.1 Variables	52
3.2.2 Cost Function	53
3.2.3 Optimization	54
3.2.4 Solution Techniques	54
3.2.5 Speedup Techniques	56
3.2.6 Discussion	59
4 Synthetic Image Test Results	60
4.1 Synthetic Images	60
4.2 Algorithm Performance	61
4.2.1 Easy Crater Images	62
4.2.2 Hard Crater Images	70
4.2.3 Hill Images	78
4.2.4 Mountain Images	86
4.3 Summary of the Results	94
4.4 Performance	94
5 Viking Image Results	96
5.1 Viking Space Project	96
5.1.1 Viking Mission	97
5.1.2 Viking Orbiter Visual Subsystem	98

CONTENTS

9

5.2	Data selection	98
5.3	Data Processing	99
5.4	Data Analysis	100
5.4.1	Example 1	100
5.4.2	Example 2	109
5.5	Summary of the Results	117
6	Error Analysis	120
6.1	Theoretical Error	120
6.2	Image Errors	124
6.3	Lighting Condition, Camera Position and the effects on DFSS results	124
6.4	Comparison between DFSS results and that of Shape from Shading .	133
7	Discussion	135
7.1	Merits	135
7.2	limitations	136
A	DFSS Functions	137
B	DFSS I/O Functions	139
C	The Matrix Library	152
C.1	Vectors and Matrices	152
C.2	Creating and Freeing Matrices	153
C.3	Accessing Elements within an Matrix	153
C.4	Common Operations on Matrices	154

This page is left blank intently.

List of Figures

1-1	Example sketch of cameras and light source in a pair of Viking images	21
1-2	Example image pair from Viking Space Project. The difference in shading is clearly shown, due to the different shading conditions . . .	22
2-1	Domain System Choices	29
2-2	Perspective Projection	30
2-3	Orthographic Projection	31
2-4	Image Formation Process	33
2-5	Image Irradiance Process	34
2-6	Stereo Geometry	37
2-7	Global Coordinate System	42
2-8	Aligned Coordinate System	44
3-1	Module Based Fusion	49
3-2	Variational methods Based Fusion	50
3-3	Centralize Algorithm Tree Based on Disparity	52
3-4	Hierarchical Basis Functions	57
4-1	Easy Crater Synthetic Images Camera Geometry. Graph shows the camera position and direction(dotted lines) and light source direction(solid lines)	63
4-2	Easy Crater Synthetic Images Camera Geometry projected in xz and yz plane	64

4-3	Easy Crater Synthetic Images	65
4-4	<i>z-only</i> DFSS iteration history on Easy Crater Images. Graph shows the <i>z-only</i> DFSS algorithm performance on the easy crater image pair. The upper graph shows the cost function(solid line) and RMS error(dashed line) of the estimated surface. The lower graph shows history of the smoothness weighting term λ	66
4-5	<i>z-only</i> DFSS Estimated Surface and Error on Easy Crater Images at the end of iterations	67
4-6	<i>z-only</i> DFSS estimation at various iteration steps on Easy Crater Images	68
4-7	<i>z-only</i> DFSS estimation at various iteration steps on Easy Crater Images	69
4-8	Hard Crater Synthetic Images Camera Geometry. Graph shows the camera position and direction(dotted lines) and light source direction(solid lines).	71
4-9	Hard Crater Synthetic Images Camera Geometry projected in xz and yz plane	72
4-10	Hard Crater Synthetic Images	73
4-11	<i>z-only</i> DFSS iteration history on Hard Crater Images. Graph shows the <i>z-only</i> DFSS algorithm performance on the hard crater image pair. The upper graph shows the cost function(solid line) and RMS error(dashed line) of the estimated surface. The lower graph shows history of the smoothness weighting term λ	74
4-12	<i>z-only</i> DFSS Estimated Surface and Error on Hard Crater Images at the end of iterations	75
4-13	<i>z-only</i> DFSS estimation at various iteration steps on Hard Crater Images	76
4-14	<i>z-only</i> DFSS estimation at various iteration steps on Hard Crater Images	77
4-15	Hill Synthetic Images Camera Geometry. Graph shows the camera position, direction(dotted lines and light source direction(solid lines).	79
4-16	Hill Synthetic Images Camera Geometry projected in xz and yz plane	80

4-17 Hill Synthetic Images	81
4-18 <i>z-only</i> DFSS iteration history on Hill Images. Graph shows the <i>z-only</i> DFSS algorithm performance on the hill image pair. The upper graph shows the cost function(solid line) and RMS error(dashed line) of the estimated surface. The lower graph shows history of the smoothness weighting term λ	82
4-19 <i>z-only</i> DFSS Estimated Surface and Error on Hill Images at the end of iterations	83
4-20 <i>z-only</i> DFSS estimation at various iteration steps on Hill Images . . .	84
4-21 <i>z-only</i> DFSS estimation at various iteration steps on Hill Images . . .	85
4-22 Mountain Synthetic Images Camera Geometry. Graph shows the camera position and direction(dotted lines) and light source direction(solid lines).	87
4-23 Mountain Synthetic Images Camera Geometry projected in xz and yz plane	88
4-24 Mountain Synthetic Images	89
4-25 <i>z-only</i> DFSS iteration history on Mountain Images. Graph shows the <i>z-only</i> DFSS algorithm performance on the mountain image pair. The upper graph shows the cost function(solid line) and RMS error(dashed line) of the estimated surface. The lower graph shows history of the constraint λ	90
4-26 DFSS Estimated Surface and Error on Mountain Images at the end of iterations	91
4-27 DFSS estimation at various iteration steps on Mountain Images . . .	92
4-28 DFSS estimation at various iteration steps on Mountain Images . . .	93
5-1 Mars Viking Images pair Camera Geometry. Graph shows the camera position and direction(dotted lines), as well as the light source direction(solid lines)	102

5-2	Mars Viking Images pair Camera Geometry projected in xz and yz plane	103
5-3	Mars Viking images pair and gradient contour	104
5-4	DFSS iteration history on Mars Viking Image pair. Top graph shows the history of cost function. Bottom graph shows the history of the smoothness weighting term λ	105
5-5	DFSS Estimated images and Surface on Mars Viking Image pair at the end of iterations	106
5-6	DFSS estimation at various iteration steps on Mars Viking Image pair	107
5-7	DFSS estimation at various iteration steps on Mars Viking Image pair	108
5-8	Mars Viking Images pair Camera Geometry. Graph shows the camera positions and directions(dotted lines), also the light source direction(solid lines).	110
5-9	Mars Viking Images pair Camera Geometry projected in xz and yz plane	111
5-10	Mars Viking Images pair and gradient contour	112
5-11	DFSS iteration history on Mars Viking Image pair. Top graph shows the history of cost function. Bottom graph shows the history of smoothness weighting term λ	113
5-12	DFSS Estimated Surface on Mars Viking Image pair at the end of iterations	114
5-13	DFSS estimation at various iteration steps on Mars Viking Image pair	115
5-14	DFSS estimation at various iteration steps on Mars Viking Image pair	116
5-15	Detailed look of the crater like feature in example two, real images and zoomed in images around the crater.	118
5-16	Detailed look of the crater like feature in example two, estimated images and surface around the crater.	119
6-1	Test Crater Synthetic Images Camera Geometry. Graph shows the camera position and direction(dotted lines) and light source direction(solid lines)	126

6-2	Test Crater Synthetic Images Camera Geometry projected in xz and yz plane	127
6-3	Test Crater Synthetic Images	128
6-4	<i>z-only</i> DFSS iteration history on Test Crater Images. Graph shows the <i>z-only</i> DFSS algorithm performance on the test crater image pair. The upper graph shows the cost function(solid line) and RMS error(dashed line) of the estimated surface. The lower graph shows history of the smoothness weighting term λ	129
6-5	<i>z-only</i> DFSS Estimated Surface and Error on Test Crater Images at the end of iterations	130
6-6	<i>z-only</i> DFSS estimation at various iteration steps on Test Crater Images	131
6-7	<i>z-only</i> DFSS estimation at various iteration steps on Test Crater Images	132
6-8	Test Crater Synthetic Image (left camera) and the estimated surface at the end of iteration.	134

This page is left blank intently.

List of Tables

3.1	Photo-Topography problem source and constraint matrix	51
4.1	Camera Geometry	61
4.2	Performance Comparison	94
5.1	Geometric Information about the first pair of Viking Images	101
5.2	Geometric Information about the first pair of Viking Images	109
A.1	DFSS function list	138

LIST OF TABLES

18

This page is left blank intently.

Chapter 1

Introduction

Machine vision is the study of algorithms and techniques for analyzing and processing visual inputs so as to determine one or more properties of the external world. Following Marr's [15] Classification, vision algorithms can be grouped into one of three levels by the processing and analyzing stage: Early vision, Intermediate vision, and High level vision. I will concentrate on Early Vision which seeks to work with raw image data and to produce an estimate of some property or properties of the 3-D world. An example in this field is Shape from Shading algorithm which estimates the shape or relative depth of an object from a gray-level image, as discussed in "Robot Vision" [8].

Machine vision is closely allied with three fields, image processing, pattern classification and scene analysis. Machine vision differs from image processing in that the result is not a better or enhanced image, not a new image, but an estimate of some external properties of the 3-D world. Pattern classification and scene analysis are associated with intermediate vision and high level vision. Unlike computer graphics which tries to produce a realistic image from a stored model of the world, machine vision tries to produce a realistic model of the world from an image. In this way, a vision system (camera+algorithm) can be described as a sensor that converts a large number (N^2 on an N -by- N grid) of measurements into a representation of the exter-

nal world, for example, height information of a N by N pixel image. Vision system is complex because it process information in 2-D in contrast to most system processing 1-D or 0-D information. Its complexity also stems from input in 2-D and the result in even higher 3-D. In another word, it is a synthesis process rather than a deduction process. We can not at this stage to build a “universal” or “general purpose” vision system. Instead, we address ourselves either to system that perform a particular task in a controlled environment or to a model that could eventually become part of a general purpose system.

In this thesis, I will describe a machine vision algorithm that combines the methods of three successful early vision algorithms, its implementation and the performance on real images, namely Viking Mars Survey images. The algorithm seeks to determine the topology of a planet from two images, which are taken at different time by two cameras at different location with different lighting conditions.

1.1 Background

Over the past two decades many early vision algorithms have been developed. Most notably, Algorithms have been developed for edge finding [16], Binocular Stereo [17, 3], Photometric Stereo [28, 20], Shape from Shading [8, 11], Shape from texture [13], Structure from Motion [25] and Optical Flow [10]. These algorithms, for the most part, are very sensitive to noise in the image(s). They seem to perform well on synthetic images, but perform poorly on real images. In order to make the algorithms more robust, some researchers are moving toward integration or fusion of two or more of these methods. The typical fusion paradigm is to explore physical constraints that are present between the solutions of the candidate methods. These constraints are then used to combine the outputs of each method to produce a fused solution to the vision problem. The hope is that by combining the information available from disparate methods, a more robust vision system can be formed.

Figure 1-1: Example sketch of cameras and light source in a pair of Viking images

The vision algorithms mentioned above fall into one of two camps: those algorithms that are based on variational formulations, such as Horn's Shape from Shading algorithm, and those algorithms that are feature based, such as the Marr and Poggio's Binocular Stereo algorithm. Variational methods usually result in an optimization problem while feature based methods usually employ direct search methods. Both approaches have been successful for certain problems.

The type of algorithm affects how easy it is to integrate or fuse more than one algorithm. In general, the feature based methods are hard to fuse since the algorithms are highly specialized and tuned to each task. On the other hand, the variational methods perhaps can be easily fused by simply combining the cost functions from disparate methods intelligently and forming a combined optimization problem. In the discussion below, I will present a method based on variational approach.

The problem on hand is to fuse the Shape from Shading, Binocular Stereo, and Photometric Stereo algorithms so as to obtain more accurate and robust estimates of the surface topography. Following its originally proposed name ¹, I call the resulting algorithm Depth from Shading and Stereo (DFSS). The research here mainly is concerned with the robustness, efficiency, and performance of the *z-only* DFSS algorithm in dealing with real images.

This research is motivated by a problem proposed by NASA. NASA is interested in determining the surface structure (or topography) of the planets in our solar system.

¹Depth from Shading and Stereo was proposed and developed by Clay Thompson in his 1992 Ph.D. thesis [24]

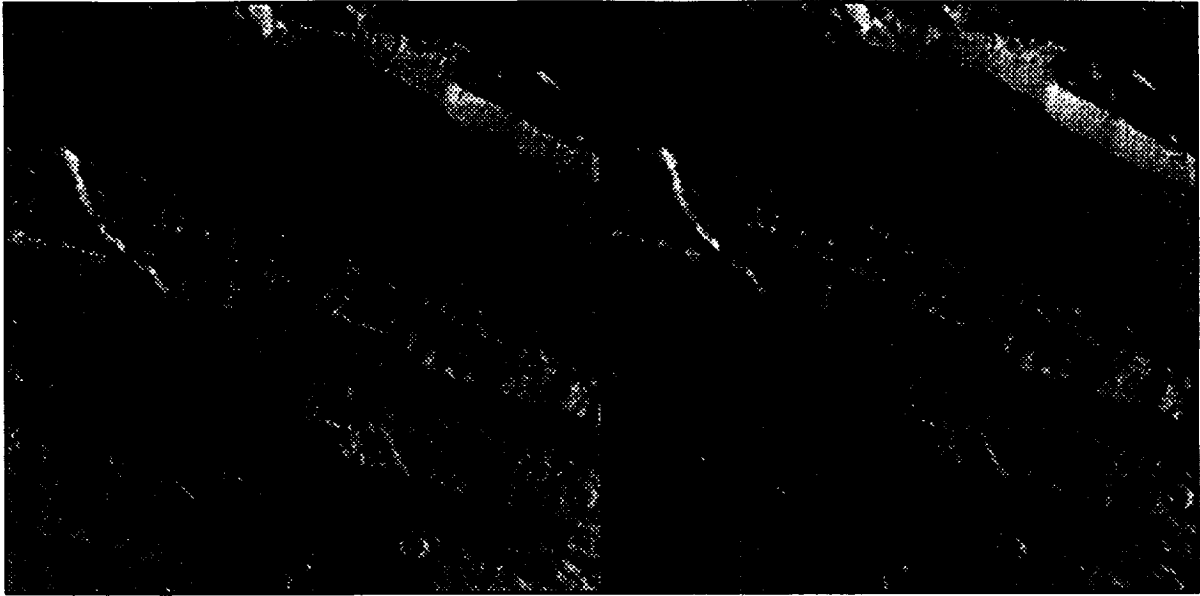


Figure 1-2: Example image pair from Viking Space Project. The difference in shading is clearly shown, due to the different shading conditions

Toward this end, NASA has used its explorer probes (e.g. Viking and Voyager) to obtain images of the same patch of surface on a planet from two different, but known, locations (see Figure 1-1). Since the images are taken at different time, the sun is not in the same position relative to the planet or the camera. This results in the images often being radically different from each other. (see Figure 1-2) This means none of the existing algorithm can handle the problem.

The importance of this property becomes clear when you compare this situation with the assumptions of several Early Vision algorithms. Binocular stereo algorithms usually assume that the two images only differ by an offset (called disparity) that is caused by the projection of a 3-D object. This implies that the images should look very similar. As mentioned above, the NASA Viking images do not meet this

assumption, hence the normal Binocular stereo algorithms will fail on these images.

Photometric stereo algorithms, on the other hand, assume that the images were taken from the same camera position but with different light positions. This implies that the corresponding points in each image are the projection of the same point in the scene. Again, the NASA Viking images do not meet this assumption, and the Photometric stereo algorithms will not work.

Only the Shape from Shading algorithms can be used with NASA Viking images, except that the images must be processed one at a time. This results in two different interpretations of the planet's surface topography.

The NASA Viking images are thus a natural choice for the investigation of fusion techniques. The Depth from Shading and Stereo algorithm I have implemented will incorporate the three modules mentioned above into one.

1.2 Planet Photo-Topography

The planet photo-topography problem in this thesis seeks to determine the topology of a planet's surface based on two images of the planet, taken from two different vantage points at two different time, as shown in Figure 1-1 and Figure 1-2. As discussed in previous section, no ready algorithm can be applied to this problem. However, it bears great resemblance with some well understood and developed algorithms, specifically, Binocular Stereo, Photometric Stereo, and Shape from Shading.

Planet photo-topography has certain similarity to Binocular Stereo. Unfortunately, it is more complicated than stereo vision in two major ways. First, The images are typically taken at two different time and position; Second, the distance between the two camera position are usually large and the camera directions are different. This means two different light source and camera positions and directions. As the result, The two images might look quite different from each other, even though they are images of the same surface patch, see Figure 1-2.

Contrast to this, Binocular image pairs are usually taken simultaneously, from positions that are near to each other, and with the same lighting. The images look very similar except for a relative shift (i.e. disparity) of objects due to their distance from cameras. If the disparity for all points in the image and the relative distance from the cameras is known, then the depth of the objects can be computed directly. The images are similar, so most stereo algorithm determine the disparity by trying to match features in one image with features in the other. Unfortunately, this approach won't work for planet photo-topography images.

Planet photo-topography also shares some aspects with Shape from Shading problem. Shape from Shading takes a gray-scale image of a surface and determines the surface topology by exploiting the Shading information in the image. Shape from Shading requires that the surface reflectance properties to be know. Assuming the reflectance properties are known, we could use a Shape from Shading algorithm to estimate the surface topology from each of the planet images. Unfortunately, the surface estimates based on each image will be different. They may not even be similar. Worst of all, the surface estimates from each image may not have the same orientation; one could be concave while the other is convex.

Planet photo-topography also has aspects in common with Photometric Stereo problem. A Photometric Stereo takes two images of the same scene with two different lighting conditions. The cameras is not moved between images. The result is two images that look different but where the correspondence is known explicitly. If the light positions are far apart enough, it is possible to determine the surface orientation directly. For Lambertian reflectance, two images can constraint the surface orientation to two possible values at each point.

The NASA Viking images are based on two different light source positions and directions. Like Photometric Stereo, the two light sources can constrain the surface orientation, if the correspondence is known. But the correspondence is based on Binocular Stereo. Thus planet photo-topography, Photometric Stereo and Binocular

Stereo are closely linked.

1.3 Viking Project, Planet Image Data

Digital image data from the Viking Mission to Mars, NASA's Planetary Data System(PDS), have been available. Through the Geosciences Discipline Node at Washington University, the Image Node at the U. S. Geological Survey, Flagstaff, Arizona and the Jet Propulsion Laboratory, the digital archive of images acquired by the Viking orbital Visual Imaging Subsystem (VIS), including the Experiment Data Record (EDR), are placed on compact read only optical disk media (CD-ROM).

1.3.1 Viking Mission

The Viking Mission consisted of four spacecraft: two identical orbiters and two identical landers. One of the orbiter experiment was the Visual Imaging Subsystem (VIS), which acquired the images that are used in this thesis.

The Viking orbiter spacecraft operated in orbit around Mars from 1976 to 1980. The orbiter imaging systems imaged all of the terrains on Mars, collected some color and stereo images, and made observations of Phobos and Deimos. Some image sequences acquired by the VIS experiment include systematic medium and high resolution coverage of large portions of the surface, stereo images, observations of Phobos and Demios, color images of the equatorial regions, observations of the polar regions, and monitoring dust storm activity.

1.3.2 Viking Orbiter Visual Subsystem

Each Viking Orbiter was equipped with two identical vidicon cameras, called the Visual Imaging Subsystem (VIS) [26], [14], [1]. Each VIS camera consisted of a telescope, a slow scan vidicon, a filter wheel, and associated electronics. A digital image was generated by scanning the vidicon face plate. A full resolution, uncompressed

Viking orbiter image consists of an array of 1056 lines with 1204 samples per line. The images then transmitted back to earth station. The images were radiometrically and geometrically calibrated and stored on tape. Subset of the images are distributed on CD-ROM.

1.4 Related Research

The algorithm discussed here is mostly related to the works of Horn [12], [9], [8], [11], Gennert [5], and Szeliski [21], [22]. Variational (least squares) approach is based significantly on the work of Horn [11], [10], [19]. The Shape from Shading part is build upon the work of Horn [11], Szeliski [22]. The stereo part is based on the gray-scale stereo algorithm of Gennert [5]. Hierarchical basis functions of Szeliski and use conjugate gradient optimization as do Leclerc and Bobick. This work also has close relation to the work of Hartt and Carlotto [6], [7], [27], and McEwen [18].

The Depth from Shading and Stereo was proposed and developed by Clay Thompson in his PH.D. thesis, [24]. In his thesis, Thompson proposed a fusion method with variational technique based on Shape from Shading and Stereo. A cost function combined Shape from Shading and stereo with smoothness regulated term is used. Conjugate gradient optimization is performed to estimate depth information. Test on synthetic images show *z-only* algorithm is the best performed and the most robust method. I will discuss the details of *z-only* algorithm in chapter 2 and chapter 3. Test on synthetic images are presented in chapter 4.

1.5 About This Thesis

In the following Chapters, I will lay the ground for Depth from Shading and Stereo by discussing each underlying fused algorithms, Shape from Shading, Binocular Stereo and Photometric Stereo, and presents *z-only* Depth from Shading and Stereo algorithm's performance on synthetic test image pairs and real Mars Viking Space Project

image pairs. In Chapter 2, I will discuss each components of DFSS algorithm, Shape from Shading, Binocular and Photometric Stereo in detail. In Chapter 3, I will show the formulation of DFSS algorithm in a variational approach. In Chapter 4, I will present the result of running DFSS algorithm on synthetic images, to understand its strength and weakness. Chapter 5 shows the performance of DFSS algorithm on Mars image pairs from Viking Space Project. Chapter 6 is devoted to analysis of various error and its effects on the algorithm's performance on real images. Chapter 7 summaries and discuss issues in DFSS' merits, limitation, implementation and its extension.

This thesis is organized for people with various backgrounds in vision field. For people who are familiar with machine vision, they can skip to chapter 3. For those who are familiar with DFSS algorithm, they can skip to chapter 5, and compare the result on Viking images with that on synthetic images.

Chapter 2

Overview

In this chapter we discuss how images are formed and how they are represented by a computer. Understanding image formation is a prerequisite for full understanding of the methods for recovering information from images. The focus will be on coordinate system, image generation process and photo-topography properties. In particular, we will discuss Binocular Stereo, photometric stereo, and shape from shading. Finally, a set of simplified equations aiming at photo-topography problem are summarized.

2.1 Coordinate System

An image is a two dimensional pattern of brightness. In analyzing the process by which a three-dimensional world is projected onto a two-dimensional image plane, we have to first set up the proper coordinate system. The most straight forward description is that the surface represents a height function over some selected 2-D domain., such as:

$$z = z(x, y) \quad (2.1)$$

Image domain can be defined in two ways, the *image-centered domain* and the *object-centered domain*. The *image-centered domain* uses the image coordinates as

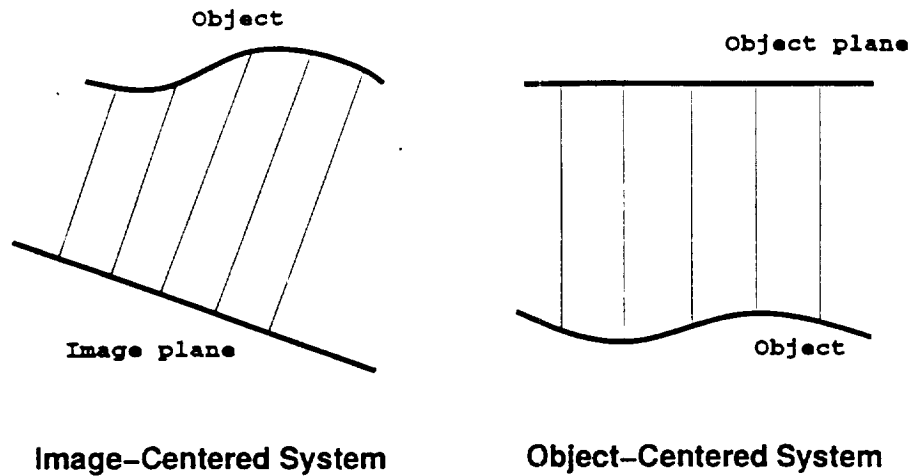


Figure 2-1: Domain System Choices

the fundamental domain and assigns the depth(or height) value to each surface point that projects to each image position. The *object-centered domain* uses the object coordinate system as the fundamental domain and assigns a surface depth(or height) information to each point in the domain. (See Figure 2-1)

Once the domain is selected, we still have the freedom to choose between *perspective* and *orthographic* projection.

2.1.1 Perspective Projection

Consider an ideal pinhole sits in between an object and an image plane (See Figure 2-2). Since the light travels in straight lines, each point in the image corresponds to a particular direction defined by a ray from the point through that pinhole. This is perspective projection.

The optical Axis thus defined to be perpendicular from the pinhole to the image plane. A Cartesian coordinate system is setup with the origin at the pinhole and *z-axis* along optical axis pointing to the image. The *z* component of the coordinates of the image will therefore be negative.

For each point P on some object in front of the camera will appear P' on the

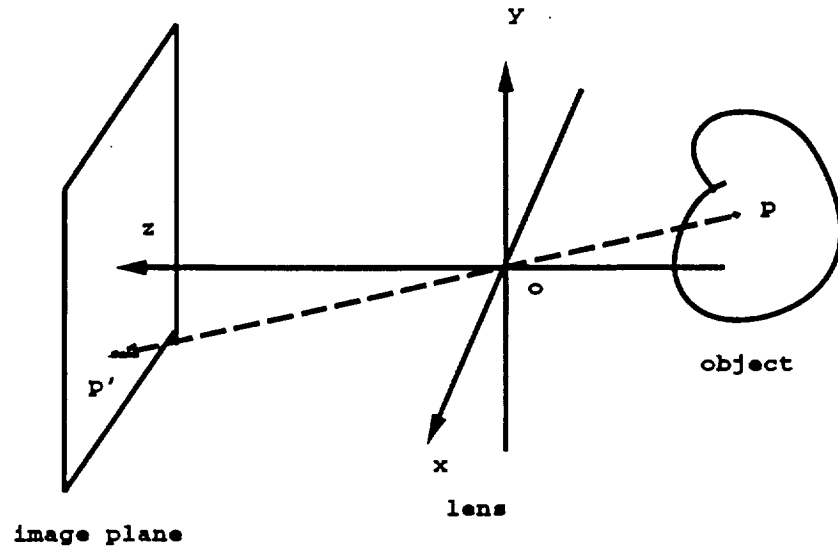


Figure 2-2: Perspective Projection

image. (See Figure 2-2). Let $\mathbf{r} = (x, y, z)^T$ denotes P , and $\mathbf{r}' = (x', y', f)^T$ ¹ denotes P' . From geometry optics, we know:

$$r = -z \sec \alpha = -(\mathbf{r} \cdot \hat{\mathbf{z}}) \sec \alpha \quad (2.2)$$

where $\hat{\mathbf{z}}$ is the unit vector along the optical axis. The length of \mathbf{r}' is

$$r' = f \sec \alpha \quad (2.3)$$

$$\frac{1}{f} \mathbf{r}' = \frac{1}{\mathbf{r} \cdot \hat{\mathbf{z}}} \mathbf{r} \quad (2.4)$$

or

¹ $z' = f$, f is the focal length.

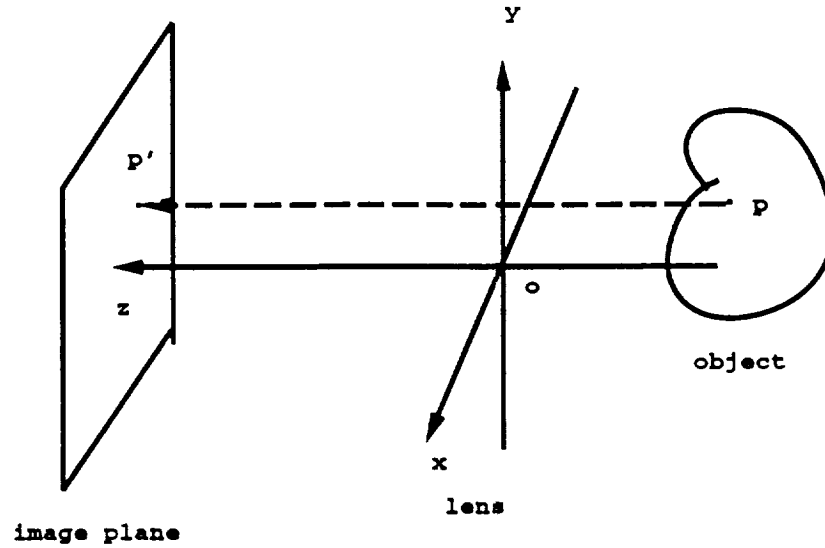


Figure 2-3: Orthographic Projection

$$\frac{x'}{f} = \frac{x}{z} \quad \text{and} \quad \frac{y'}{f} = \frac{y}{z} \quad (2.5)$$

2.1.2 Orthographic Projection

Consider that if we put the image plane at $z = z_0$, and define *lateral magnification* m as the ratio of the distance between two points measured in the image to the distance between the corresponding points on the plane.

$$m = \frac{f}{-z_0} \quad (2.6)$$

where $-z_0$ is the distance of the plane from the pinhole.

A small object at an average distance $-z_0$ will give rise to an image which is magnified by m . Let the *depth* be the distance from the object to the camera. The magnification is approximately constant when the *depth* range of the scene is relative

small to the average distance. Then

$$x' = -mx \quad \text{and} \quad y' = -my \quad (2.7)$$

For convenience, we can set $m = -1$, and

$$x' = x \quad \text{and} \quad y' = y \quad (2.8)$$

This is the orthographic projection. It can be depicted as the ray runs parallel to the optical axis (See Figure 2-3).

If we choose *image-centered* domain, and orthographic projection is used, the projection map is straightforward. However, if perspective projection is used, then this mapping can become quite complex, for example, surface normal calculation. If we choose *object-centered* domain, both perspective and orthographic projection are straightforward. However, the projected points in general won't map to the center of each pixel. Thus interpolation is needed to obtain values at each pixel.

2.2 Shape From Shading

2.2.1 Image Formation Process

Shape from Shading problem is to generate three dimensional topography information from two dimensional image. It is crucial to understand how images are formed. This process can be viewed as two stages, object radiance and image formation. This process also depends on four factors:

1. object irradiance.
2. reflectance map.

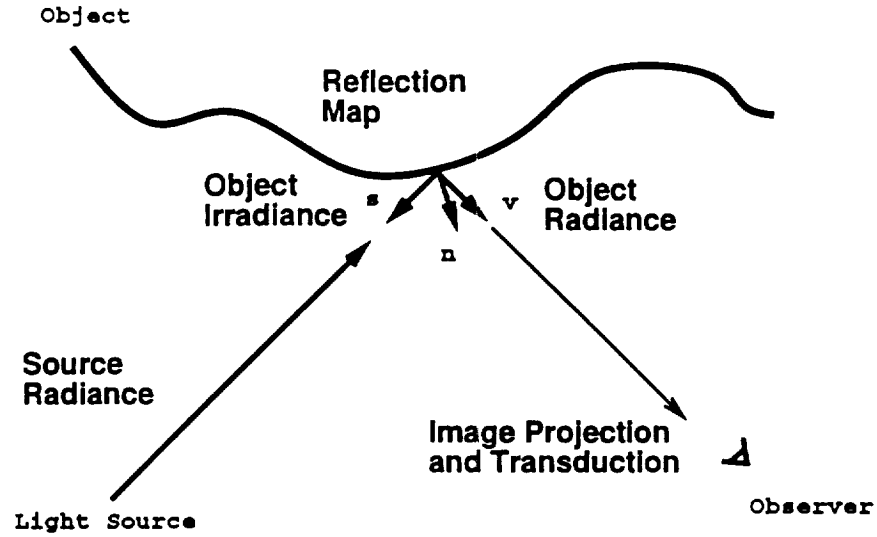


Figure 2-4: Image Formation Process

3. image projection.
4. image transduction.

2.2.2 Object radiance stage

The amount of light falling on a surface is called the *irradiance*. At each point ξ on the surface, and for each direction \hat{s} , the irradiance distribution function is $E(\xi, \hat{s})$. The amount of light radiated from a surface is called *radiance*. Obviously, the object radiance depends on the object irradiance and the surface reflectance properties. The surface reflectance properties is the physical character of the surface, independent of the irradiance, and can be described by *Bidirectional Reflectance Distribution Function* (BRDF).² At the point ξ , the BRDF $f(\xi, \hat{n}, \hat{v}, \hat{s})$ relates the brightness of the surface patch with normal \hat{n} illuminated from the direction \hat{s} and as seen from the direction \hat{v} . Using these two distribution functions, the radiance function can be written as:

²For more information on BRDF, see[8, p. 209]

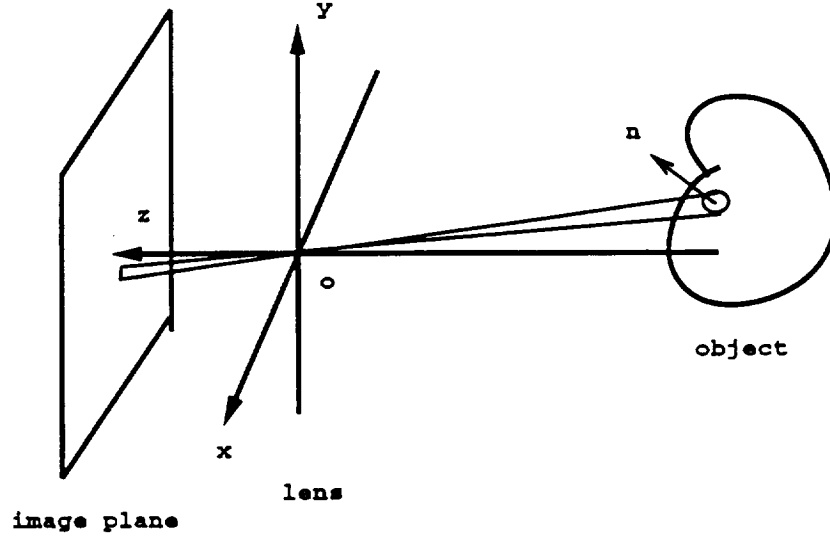


Figure 2-5: Image Irradiance Process

$$L(\xi, \hat{n}, \hat{v}) = \int \int_H f(\xi, \hat{n}, \hat{v}, \hat{s}) E(\xi, \hat{s}) (\hat{s} \cdot \hat{n}) d\omega(\hat{s}) \quad (2.9)$$

where H is the hemisphere of possible light sources directions for the patch at surface point ξ , and $d\omega(\hat{s})$ is the solid angle subtended in the direction \hat{s} . Representing \hat{s} in spherical coordinates, above equation becomes

$$L(\xi, \hat{n}, \hat{v}) dz = \int_{-\pi}^{\pi} \int_0^{\pi/2} f(\xi, \hat{n}, \hat{v}, \hat{s}) E(\xi, \hat{s}) \sin \theta \cos \theta d\theta d\phi \quad (2.10)$$

Two common reflectance models are the *Lambertian* model and the specular model. The *Lambertian* model deals matter surface. An ideal *Lambertian* surface is one that appears equally bright from all viewing direction and reflects all incident light, which means the BRDF is independent of source direction \hat{s} , surface normal direction \hat{n} and viewing direction \hat{v} . Follow this definition, we get

$$f_{Lambertian}(\xi, \hat{n}, \hat{v}, \hat{s}) = \frac{1}{\pi} \rho(\xi) \quad (2.11)$$

where $\rho(\xi)$ is the surface albedo, or the fraction of light re-emitted by the surface. Evaluate 2.10 for a *Lambertian* surface illuminated by a point source at infinity, the surface radiance is

$$L_{Lambertian}(\xi, \hat{\mathbf{n}}, \hat{\mathbf{v}}) = \frac{\lambda \rho(\xi)}{\pi} (\hat{\mathbf{s}}_0 \cdot \hat{\mathbf{n}}) \quad (2.12)$$

Specular model deals with metallic surfaces. All light from the direction $\hat{\mathbf{s}}_0$ is reflected into the direction $2(\hat{\mathbf{n}} \cdot \hat{\mathbf{s}}_0)\hat{\mathbf{n}} - \hat{\mathbf{s}}_0$, so that the BRDF is

$$f_{Specular}(\xi, \hat{\mathbf{n}}, \hat{\mathbf{v}}, \hat{\mathbf{s}}) = \rho(\xi) \delta(\hat{\mathbf{v}}, 2(\hat{\mathbf{n}} \cdot \hat{\mathbf{s}}_0)\hat{\mathbf{n}} - \hat{\mathbf{s}}_0) \quad (2.13)$$

For a point source at infinity, the surface radiance for a specular surface is found to be

$$L_{Specular}(\mathbf{R}, \hat{\mathbf{n}}, \hat{\mathbf{v}}) = \lambda \rho(\xi) \delta(\hat{\mathbf{v}}, 2(\hat{\mathbf{n}} \cdot \hat{\mathbf{s}}_0)\hat{\mathbf{n}} - \hat{\mathbf{s}}_0) \quad (2.14)$$

A radiance function representing most surface is a linear combination of these two plus a constant ambient term, which models haze and atmospheric reflection, as well as the effects of a uniformly distributed light source

$$L = \alpha L_{Lambertian} + \beta L_{specular} + \gamma \quad (2.15)$$

where α , β and γ are scalars so that $\alpha + \beta + \gamma = 1$.

Many vision researchers prefer to use a representation of the surface radiance based on a global coordinate system. This is called the Reflectance function. Given Known surface properties and a known light source, the reflectance function, $R(\xi, \hat{\mathbf{n}}, \hat{\mathbf{v}})$ is defined using the corresponding surface radiance via a change in coordinates,

$$R(\xi_G, \hat{\mathbf{n}}_G, \hat{\mathbf{v}}_G) = L(\xi, \hat{\mathbf{n}}, \hat{\mathbf{v}}) \quad (2.16)$$

where ξ_G , $\hat{\mathbf{n}}_G$, and $\hat{\mathbf{v}}_G$ are the surface position, normal direction vector, and view-

ing direction vector.

2.2.3 Image Generation Stage

In earlier section, we introduced the perspective and orthographic projection. We also need to know how the brightness of the image is affected by the projection. Assuming perfect lens, the irradiance from a patch on object is

$$E(\mathbf{r}) = L(\xi(\mathbf{r}), \hat{\mathbf{n}}, \hat{\mathbf{v}}(\mathbf{r})) \frac{\pi}{4} \left(\frac{d}{f}\right)^2 \cos \alpha^4 \quad (2.17)$$

where $L(\xi(\mathbf{r}), \hat{\mathbf{n}}, \hat{\mathbf{v}}(\mathbf{r}))$ is the radiance of the corresponding object patch, d is the diameter of the lens, and α is the off-axis angle of the projecting ray. Here, $\hat{\mathbf{v}}$ and ξ are functions of \mathbf{r} via the projection from image points \mathbf{r} to object points. Equivalently, this equation can be restated using reflectance function instead,

$$E(\mathbf{r}) = R(\xi(\mathbf{r}), \hat{\mathbf{n}}, \hat{\mathbf{v}}(\mathbf{r})) \frac{\pi}{4} \left(\frac{d}{f}\right)^2 \cos \alpha^4 \quad (2.18)$$

There is one more factor in image generation stage. Light will be converted into digital signal, during the conversion, there is distortion of the image. The effect can be removed by calibration so that the measured image irradiance can be related to object radiance in a straightforward way. With a perfect calibration, the image irradiance equation can be put into a very simple form,

$$E(\mathbf{r}) = R(\xi(\mathbf{r}), \hat{\mathbf{n}}, \hat{\mathbf{v}}(\mathbf{r})) \quad (2.19)$$

where the constant term and $\cos \alpha^4$ can be set to 1 during calibration.

2.3 Stereo

For normal stereo situations, the camera are close together and both pictures are taken simultaneously or nearly so. The stereo images that result look very similar,

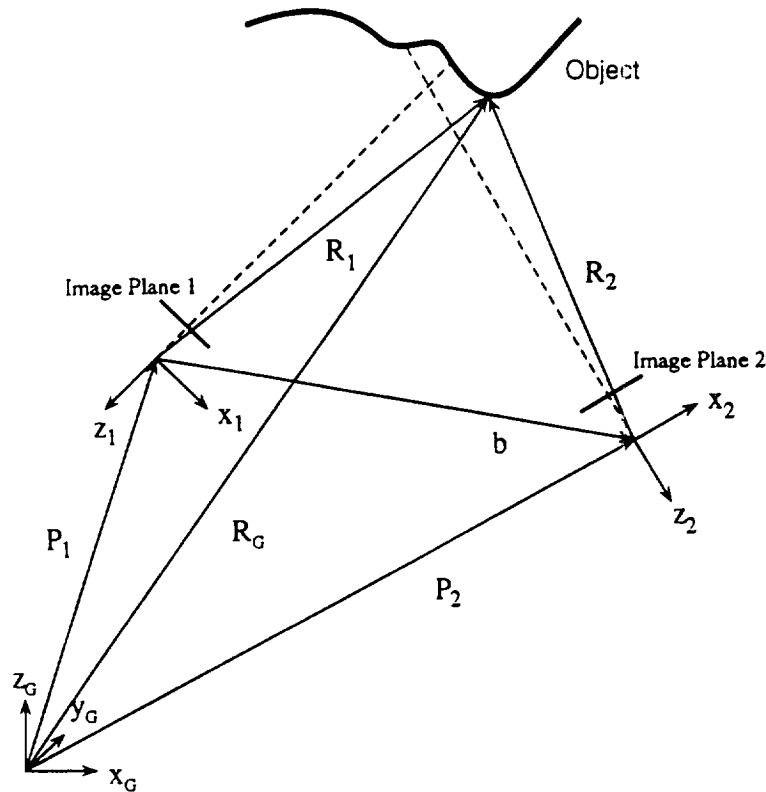


Figure 2-6: Stereo Geometry

mostly differing in a shift of objects in each image caused by perspective projection. the difference in the shift of an object point in the left image and the right image is called the *disparity*. If the relative position of the cameras is known, and it is known the correspondence relation of each pixel in the right and left image, it is possible to determine the depth directly for the surface points that projects to those pixels.[8]

To determine the depth directly from the disparity, See Figure 2-6. If the corresponding points in each image map to rays intersect, we can use geometry to determine the depth of the point that is halfway between the rays at their closest approach.

First find the relationship between the two camera coordinate system, Suppose, we know the position of the principal point of each camera in some global coordinate system, P_1 and P_2 , and we also know the rotational transformation matrices from each local camera coordinate system to the global coordinate system, T_1 and T_2 , the coordinates of the point ξ in the two camera coordinate system is

$$\begin{aligned}\mathbf{R}_1 &= T_1^{-1}(\xi - \mathbf{P}_1) \\ \mathbf{R}_2 &= T_2^{-1}(\xi - \mathbf{P}_2)\end{aligned}\tag{2.20}$$

By moving ξ from the above equations and defining $\mathbf{b} = \mathbf{P}_2 - \mathbf{P}_1$, the relationship between a point in Camera Coordinate System 1 and a point in Camera Coordinate System 2 is

$$\mathbf{R}_1 = T_1^{-1}\mathbf{b} + T_1^{-1}T_2\mathbf{R}_2\tag{2.21}$$

Now, determine the relationship between disparity and depth. Suppose we are given image points, $\mathbf{r}_1 = (x_1, y_1, f)^T$ and $\mathbf{r}_2 = (x_2, y_2, f)^T$ (one in each image), that correspond to the same surface point, then the best estimate for the surface position can be found by finding the point on each ray (along \mathbf{r}_1 and \mathbf{r}_2) where the distance between the ray is minimized. That is the problem

$$\min_{s,t} \|\mathbf{b} + t\mathbf{r}_2 - s\mathbf{r}_1\|^2\tag{2.22}$$

must be solved where s and t are scalar parameters. For now assume all the vectors are given on the same basis.

By differentiating the above equation with respect to s and t , setting the resulting equation to zero, and solving, it is found that the minimum occurs when

$$\begin{aligned}s &= \frac{(\mathbf{r}_1 \cdot \mathbf{r}_2)(\mathbf{b} \cdot \mathbf{r}_1) - (\mathbf{r}_1 \cdot \mathbf{r}_2)(\mathbf{b} \cdot \mathbf{r}_2)}{(\mathbf{r}_1 \cdot \mathbf{r}_1)(\mathbf{r}_2 \cdot \mathbf{r}_2) - (\mathbf{r}_1 \cdot \mathbf{r}_2)^2} = \frac{(\mathbf{r}_2 \times \mathbf{b}) \cdot (\mathbf{r}_2 \times \mathbf{r}_1)}{\|\mathbf{r}_2 \times \mathbf{r}_1\|} \\ t &= \frac{(\mathbf{r}_1 \cdot \mathbf{r}_2)(\mathbf{b} \cdot \mathbf{r}_1) - (\mathbf{r}_1 \cdot \mathbf{r}_2)(\mathbf{b} \cdot \mathbf{r}_2)}{(\mathbf{r}_1 \cdot \mathbf{r}_1)(\mathbf{r}_2 \cdot \mathbf{r}_2) - (\mathbf{r}_1 \cdot \mathbf{r}_2)^2} = \frac{(\mathbf{r}_1 \times \mathbf{b}) \cdot (\mathbf{r}_2 \times \mathbf{r}_1)}{\|\mathbf{r}_2 \times \mathbf{r}_1\|}\end{aligned}\tag{2.23}$$

if \mathbf{r}_1 , \mathbf{r}_2 and \mathbf{b} are coplanar then the above set of equations is just a fancy repre-

sensation of the sines.

The global position of the point halfway between the two rays at their closest approach is

$$\begin{aligned}\xi &\approx \mathbf{P}_1 + s\mathbf{r}_1 + \frac{1}{2}(\mathbf{b} + t\mathbf{r}_2 - s\mathbf{r}_1), \\ &= \mathbf{P}_1 + \frac{1}{2}\mathbf{b} + \frac{1}{2} \frac{(T_2\mathbf{r}_2 \times T_1\mathbf{r}_1)}{\|T_1\mathbf{r}_2 \times T_2\mathbf{r}_1\|^2} [(T_2\mathbf{r}_2 \times \mathbf{b})T_1\mathbf{r}_1 + (T_1\mathbf{r}_1 \times \mathbf{b})T_2\mathbf{r}_2] \quad (2.24)\end{aligned}$$

where \mathbf{P}_1 , \mathbf{P}_2 and \mathbf{b} are given in global coordinates and \mathbf{r}_1 , \mathbf{r}_2 are given in the appropriate local camera coordinate system.

The equations simplify greatly if \mathbf{r}_1 , \mathbf{r}_2 and \mathbf{b} are coplanar, the cameras are aligned so that the optical axes are in the direction $-\hat{\mathbf{z}}$ and $\hat{\mathbf{x}}$ is along \mathbf{b} . The stereo geometry is commonly assumed to exist for most binocular stereo algorithms. With these restrictions, the above equation becomes

$$\xi = \mathbf{P}_1 + \frac{b}{(x_2 - x_1)}\mathbf{r}_1 \quad (2.25)$$

where \mathbf{r}_1 and \mathbf{r}_2 are defined as earlier, and $\mathbf{b} = (b, 0, 0)^T$. the quantity $(x_2 - x_1)$ in the above equation is the disparity mentioned earlier. Note that the disparity can be mapped directly into depth only in the special case. For more general case, 2.24 must be used.

2.4 Photo-Topography

Now we have described image formation process and stereo system, we can formulate the photo-topography problem.

What we know from the introduction of Viking Project in previous chapter, there are two images of an area on the planet surface, taken at two different times from two different positions and angles. The objective is to determine the topography of

the planet surface.

The images in a photo-topography problem are taken from two different vantage points (See Figure 1-1). The cameras are often far apart and has pictures that are taken at different time. In this case, it is very likely for the two images to look very different. this makes the correspondence problem very hard. As opposing the small baseline disparity and the same position and time of two images.

The solution is constrained by geometry and the image generation process. Specifically each image is constrained by the perspective projection equation, 2.4 and the image irradiance equation, 2.19 and the stereo constrained equations 2.20.

Combining these equations we find that the photo-topography problem is constrained such that

$$\begin{aligned} E^{(1)}(\mathbf{r}_1) &= R^{(1)}(\xi, \hat{\mathbf{n}}, -T_1\mathbf{r}_1) \\ E^{(2)}(\mathbf{r}_2) &= R^{(2)}(\xi, \hat{\mathbf{n}}, -T_2\mathbf{r}_2) \end{aligned} \quad (2.26)$$

where

$$\begin{aligned} \mathbf{r}_1 &= \frac{fT_1^{-1}(\xi - \mathbf{P}_1)}{T_1^{-1}(\xi - \mathbf{P}_1) \cdot T_1\hat{\mathbf{z}}_1} \\ \mathbf{r}_2 &= \frac{fT_2^{-1}(\xi - \mathbf{P}_2)}{T_2^{-1}(\xi - \mathbf{P}_2) \cdot T_2\hat{\mathbf{z}}_2} \end{aligned} \quad (2.27)$$

$E^{(1)}$ and $E^{(2)}$ are the image brightness measured in the first and the second cameras respectively, and $R^{(1)}$ and $R^{(2)}$ are the reflectance maps based on the first and second light source positions. As before, ξ is the surface position in global coordinates.

It is important to review the assumptions behind these equations. The perspective projection equations assume perfect lenses and perfect knowledge of the camera principal points and optical axes. the surface radiance equation assumes we have

perfect knowledge of the surface reflectance properties, light source directions, and all surface points as visible from both cameras(i.e., there is no self occlusions). The simplified form of the image irradiance equation assumes we either have a perfect sensor or we can perfectly calibrate the sensor to remove any abnormalities from the sensor & lens combination. The stereo equations assume we know the relative position and orientation of the two camera perfectly. the only assumption that are truly artificial are the assumptions of perfect knowledge of the reflectance maps and the the assumption of no self-illumination. With more careful measurements and more expensive equipment, it is possible to approach perfect knowledge of other assumptions. The assumption that all surface points be visible merely restricts the roughness of the surface that this research is applicable to.

2.5 Simplification

There are several simplifications that can be made to the equation in the previous section to make them easier to solve.

2.5.1 Special Global Coordinate System

So far all the equations have been written for any global coordinate system. I would like to restrict the equations to a particular global coordinate system, as shown in Figure 2-7. This coordinate system is defined as below:

1. Place the origin of the global coordinate system half way between the principal points of the two cameras.
2. Choose the \hat{x}_G direction along the line connecting the two cameras. $\hat{x}_G = \mathbf{b}/\|\mathbf{b}\|$.
3. Choose the \hat{z}_G in the direction of the average of the optical axis directions of the two cameras projected into the plane perpendicular to \mathbf{x}_G , then

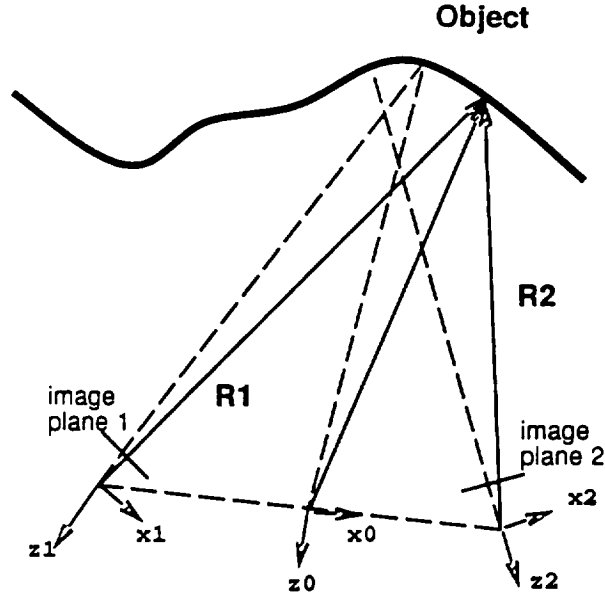


Figure 2-7: Global Coordinate System

$$\hat{\mathbf{z}}_G = \frac{(\hat{\mathbf{z}}_1 - \hat{\mathbf{z}}_1 \cdot \hat{\mathbf{x}}_G) + (\hat{\mathbf{z}}_2 - \hat{\mathbf{z}}_2 \cdot \hat{\mathbf{x}}_G)}{\|(\hat{\mathbf{z}}_1 - \hat{\mathbf{z}}_1 \cdot \hat{\mathbf{x}}_G) + (\hat{\mathbf{z}}_2 - \hat{\mathbf{z}}_2 \cdot \hat{\mathbf{x}}_G)\|}. \quad (2.28)$$

4. Choose $\hat{\mathbf{y}}_G$ in the direction of $\hat{\mathbf{z}}_G \times \hat{\mathbf{z}}_G$ in order to create a right handed coordinate system.
5. Also set up a virtual image plane with $f = 1$.

Then $\mathbf{P}_1 = -b/2$, $\mathbf{P}_2 = b/2$, and $\mathbf{b} = (b, 0, 0)^T$.

2.5.2 Removing the View Direction Dependence

A common simplification for computer vision is the assumption of a *Lambertian* reflectance map. Since a *Lambertian* surface reflects light equally in all directions, we see from Equation 2.12 that the radiance function does not depend on viewing direction. Thus the dependence on $\hat{\mathbf{v}}$ can be removed from all equations.³

³This is true for any reflectance function which is viewing independent, not just *Lambertian* reflectance.

Note that the viewing direction could also have been removed from the equations by using orthographic projection for the reflectance function while retaining perspective projection for everything else. In this case the viewing direction would be constant for each camera and its effect could be subsumed into the reflectance map. Doing this would, of course, introduce an error into the calculations since orthographic and perspective projection do not produce the same viewing direction in general. This error would be small for planet photo-topography since the cameras are so far away from the surface. The large viewing distance results a very small relative depth change $\Delta Z/Z_0$. This is especially true when the field of view is small.

2.5.3 Constant Albedo

So far the equations have terms that denote position on the surface ξ . The main reason for this dependence is to take into account varying albedo, varying reflectance properties or both. To simplify, we can assume that the reflectance properties, albedo, or both are constant across the surface. Assuming the reflectance but not albedo, are constant across the surface results in a reflectance function that is separated,

$$R(\xi, \hat{n}, \hat{v}) = \rho(\xi) \bar{R}(\hat{n}, \hat{v}) \quad (2.29)$$

where $\bar{R}(\hat{n}, \hat{v})$ is the reflectance function for a surface with uniform albedo, no interreflection, and no mutual occlusion. As for R , any light source effects are included in \bar{R} . When both the reflectance and albedo are constant, the dependence of the reflectance map on surface position can be removed,

$$R(\xi, \hat{n}, \hat{v}) = \bar{R}(\hat{n}, \hat{v}) \quad (2.30)$$

Combined with either *Lambertian reflectance*⁴ or orthographic projection, the dependence on \hat{v} can be removed,

⁴Or any viewing independent reflectance function

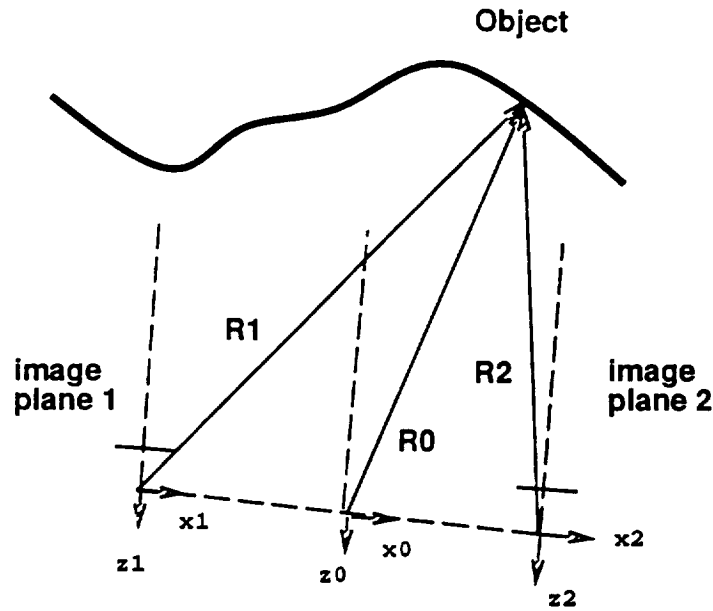


Figure 2-8: Aligned Coordinate System

$$R(\xi, \hat{n}, \hat{v}) = \bar{R}(\hat{n}) \quad (2.31)$$

The simplification restricts the applicability of the algorithm presented in next chapter so that only uniformly colored surface patches have the possibility of being estimated correctly. When algorithms based on this simplification are applied to images that violate these simplifications, we would expect errors at the transition between different colored parts of the surface, within differently colored areas or both.

2.5.4 Aligned Cameras

The Final simplification that can be made is to align the cameras so that their optical axes are parallel, which will also be parallel to the global coordinate system, As shown in Figure 2-8. When the cameras are aligned, the rotational transformations T_1 and T_2 are identity transformations, which simplifies the stereo equations to

$$\begin{aligned}\mathbf{R}_1 &= \xi - \mathbf{P}_1, \\ \mathbf{R}_2 &= \xi - \mathbf{P}_2.\end{aligned}\tag{2.32}$$

While this situation is very unrealistic for the photo-topography problem, any set of images can be re-projected into this coordinate system.

2.5.5 Summary of Simplified equations

The rest of the thesis is based on equations that take into account all the above simplifications. in summary, they are:

1. A special global coordinate system that is halfway between the two camera position.
2. All surface points are assumed to be visible from the two cameras.
3. The reflectance properties of the surface are assumed to be constant and *Lambertian* allowing the viewing direction dependence to be dropped from the reflectance equations. In addition, it is assumed that there is no interreflection between different parts of the surface.
4. The surface is assumed to have constant albedo allowing the position dependent term to be dropped.
5. The cameras optical axis are assumed to be aligned with each other allowing the rotational transformation to be set to identity.

Apply all the above, the set of equations are:

$$E^{(1)}(\mathbf{r}_1) = R^{(1)}(\hat{\mathbf{n}})$$

$$E^{(2)}(\mathbf{r}_2) = R^{(2)}(\hat{\mathbf{n}}) \quad (2.33)$$

where

$$\begin{aligned} \mathbf{r}_1 &= \frac{f(\xi + \mathbf{b}/2)}{(\xi + \mathbf{b}/2) \cdot \hat{\mathbf{z}}_1}, \\ \mathbf{r}_2 &= \frac{f(\xi - \mathbf{b}/2)}{(\xi - \mathbf{b}/2) \cdot \hat{\mathbf{z}}_2}, \end{aligned} \quad (2.34)$$

If we define $z = \xi \cdot \hat{\mathbf{z}}_G$ and $\mathbf{r} = f\xi/z$, then the constraint function can be written as

$$\begin{aligned} E^{(1)}(\mathbf{r} + \frac{f\mathbf{b}}{2z}) &= R^{(1)}(\hat{\mathbf{n}}) \\ E^{(2)}(\mathbf{r} - \frac{f\mathbf{b}}{2z}) &= R^{(2)}(\hat{\mathbf{n}}) \end{aligned} \quad (2.35)$$

It is found more convenient in subsequent chapters to use a slightly different notation. Use explicitly $\mathbf{r} = (x, y, f)$ and the normal vector $\hat{\mathbf{n}}$ is parameterized using gradient component p and q ,

$$\hat{\mathbf{n}} = \frac{(-p, -q, 1)}{\sqrt{p^2 + q^2 + 1}} \quad (2.36)$$

where

$$\begin{aligned} p &= \frac{fz_x}{xz_x + z} \\ q &= \frac{fz_y}{yz_y + z} \end{aligned} \quad (2.37)$$

The photo-topography equations then become

$$\begin{aligned}
E^{(1)}\left(x + \frac{fb}{2z}, y\right) &= R^{(1)}(p, q) \\
E^{(2)}\left(x - \frac{fb}{2z}, y\right) &= R^{(2)}(p, q)
\end{aligned} \tag{2.38}$$

2.6 Camera Calibration

In order to relate positions in the image direction vectors in 3-D space, the origin of the camera coordinate system must be known. Finding this origin is the classical *interior orientation* problem. As assumed in section 2.2.3, this origin is the projection of the principal point in the image plane. In the special coordinate system, see Figure 2-8, the position of each camera coordinate system origin can be specified by a vector $\mathbf{v} = (v_x, v_y, f^T)$. these vector specify the offset of the camera coordinate origin for each image. Suppose \mathbf{v}_0 is the offset to an object point in the global coordinate system, then the offset to this same point in the camera images are

$$\mathbf{v}_1 = \mathbf{v}_0 + \frac{fb}{2z_0}, \tag{2.39}$$

$$\mathbf{v}_2 = \mathbf{v}_0 - \frac{fb}{2z_0}. \tag{2.40}$$

the values of \mathbf{v}_1 and \mathbf{v}_2 can be quite large in the aligned coordinate system indicating that the images must be shifted far away from the camera coordinate system origin. While this is not possible physically, it is a consequence of re-projecting real images into the aligned coordinate system.

Chapter 3

Fusion of Shape from Shading and Stereo

In this chapter, I will discuss the Fusion strategy proposed by Clay Thompson, in his Ph.D. Thesis(See [24]). I will focus on the most efficient and robust *z-only* algorithm in solving photo-topography problem. The basic idea is to closely couple the solution of Shape-from-Shading and stereo in a variational approach. The important point is to use each algorithms strength to compensate each one's weakness. (see table 3.1)

Naturally, one way of fusing two or more algorithms is to run each fused algorithms separately on the image, and then combine the output to generate a single solution. (See Figure 3-1) This approach is easy to understand and implement, since existing algorithm can be put together in a *ad-hoc* manner, but it ignores the interconnection embedded in the algorithms and the information coupling each algorithm.

The variational approach, on the other hand, closely couples the algorithms together. (See Figure 3-2) This can be achieved by formulating a combined cost function based on the cost function of each fused algorithms. The result is a combined optimization problem which takes into account both the explicit and implicit constraints between the methods. Variational methods, by their nature, can exploit any orthogonalities in the methods. It then has the potential to create robust, well performing

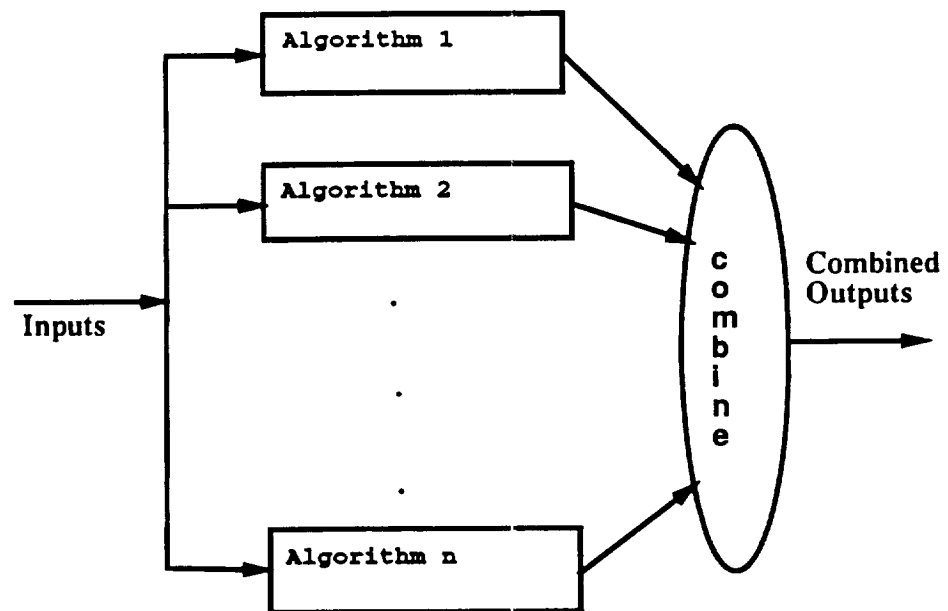


Figure 3-1: Module Based Fusion

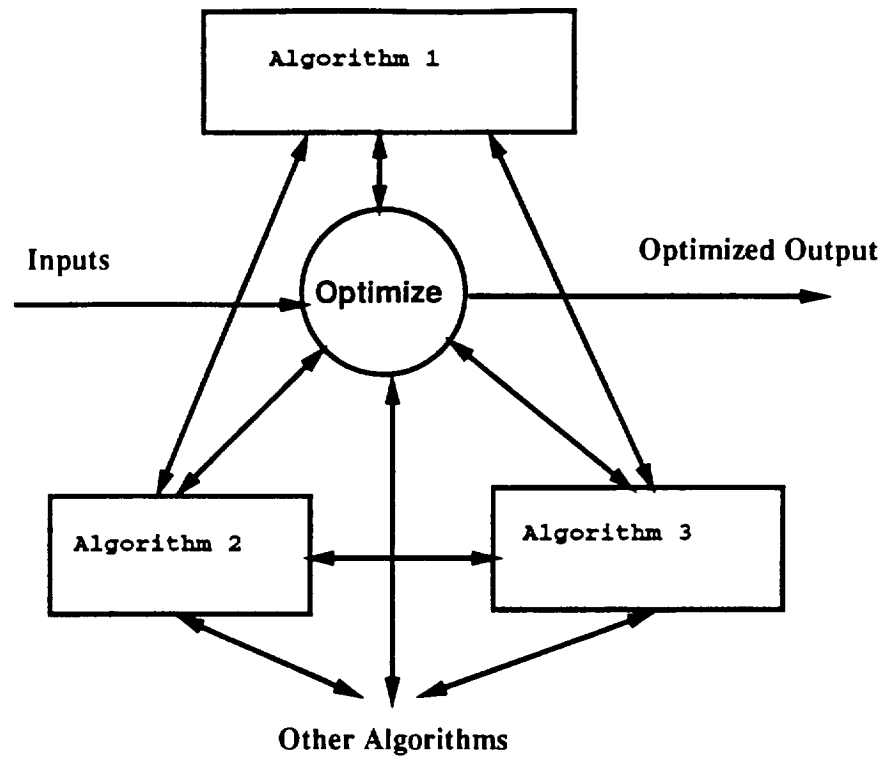


Figure 3-2: Variational methods Based Fusion

combinations of algorithms which can be applied to a wide range of input images.

3.1 Fusion Strategy

The planetary images, particularly Viking images provide us two sources of information.

1. **Shading Information:** the gray levels in each image are an indication of the surface orientation with respect to the light source.
2. **Stereo Information:** assuming corresponding pixel in each image can be matched up, the stereo information can be used to recover the shape.

	Shading	Stereo	Lighting
Shape From Shading	Shape From Shading	Surface Constraint	Surface Orientation Constraint
Binocular Stereo	Correspondence Constraint	Binocular Stereo	Correspondence Constraint
Photometric Stereo	Correspondence Constraint	Correspondence Constraint	Photometric Stereo

Table 3.1: Photo-Topography problem source and constraint matrix

3. Photometric Information: the gray levels of corresponding pixel constrain the set of possible surface orientation, since the images are taken at with two different light source position.

Another important point is that the shading and stereo information are independent and mutual compensating. independent means we can differentiate the contribution from each source of information. For example, The shading information is the strongest when shading is smooth, while stereo information is the strongest near surface discontinuities, where feature dominates, and when cameras are widely apart. The photometric information is strongest when the light source positions are widely separated.

3.2 *z-only* DFSS Algorithm

z-only algorithm estimates everything in a single global coordinate system which is defined to be half way between the two camera positions. Figure 3-3 shows the tree diagram of the flow of this algorithm. The current estimate of the surface height z is used to project points in the global coordinate system to points in each image using perspective projection. These points won't in general land on a pixel center so some type of interpolation is used to determined the value of the image at the projected points. This interpolated image F is then compared to a computed image based on

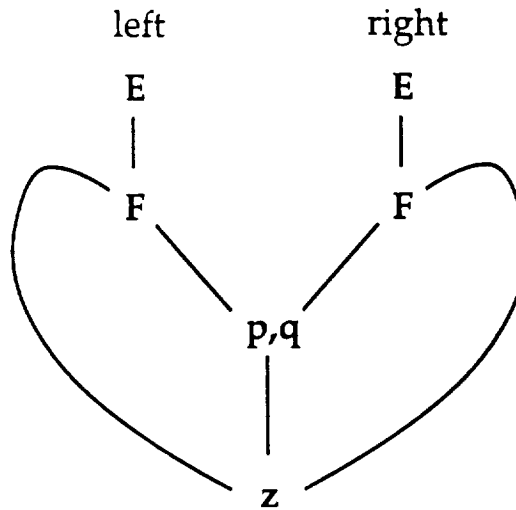


Figure 3-3: Centralize Algorithm Tree Based on Disparity

the current estimate of the surface. The error is used to update p , q , and ultimately z .

z -only algorithm uses hard integrability constraints. With hard integrability we are guaranteed that any solution obtained will be feasible. The trade-off is that z -only algorithm will have less degree of freedom and more susceptible to local minima.

3.2.1 Variables

In z -only algorithm, the depth map z is the only optimization variable, thus its name z -only algorithm. The surface gradient components p and q are computed directly from the depth map. The photo-topography images, camera geometry and surface reflectance function are inputs to the cost function.

3.2.2 Cost Function

The cost function is formed by integrating the squared photo-topography error introduced by the current estimate for z , together with a penalty function for departure from smoothness.

The penalty function is mainly used to guide the solution towards the minimum. In practice, the smoothness weighting parameter λ is slowly reduced towards zero as the algorithm converges.

$$\begin{aligned} \min_z J = & \frac{1}{2} \int \int [(E^{(1)}(x + \frac{fb}{2z}, y) - R^{(1)}(p, q))^2 + (E^{(2)}(x - \frac{fb}{2z}, y) - R^{(2)}(p, q))^2 \\ & + \lambda(z_{xx}^2 + 2z_{xy}^2 + z_{yy}^2)] dx dy. \end{aligned} \quad (3.1)$$

The smoothness term is based on the second variation. It is equivalent to $p_x^2 + p_y^2 + q_x^2 + q_y^2$ when $z_x \approx fp/z_0$ and $z_y \approx fp/z_0$ where z_0 is the nominal depth¹.

The cost function above is continuous and must be discretized before it can be optimized. The process is an approximation one. using finite difference methods, since the images are in digital form, each pixel represents the average of the brightness falling within the sensitive area of the corresponding photosensor. It then makes sense to approximate the values for p , q and z as arrays of gradient components or surface depth.

Using an array of z , the cost function can be written as:

$$\begin{aligned} \min_z J = & \frac{1}{2NM\epsilon^2} \sum_{x,y \in D} [(F^{(1)}(x, y) - R^{(1)}(p, q))^2 + (F^{(2)}(x, y) - R^{(2)}(p, q))^2 \\ & + \lambda(z_{xx}^2 + 2z_{xy}^2 + z_{yy}^2)] dx dy. \end{aligned} \quad (3.2)$$

¹The approximations are valid when the field of view is small, the image is centered around the camera's principal point, and the depth of field relative to the nominal depth is small.

where D is the discrete image domain of the underlying variables in the global coordinate system, N and M are the row and column dimensions of the discrete domain, and ϵ is the grid spacing (assume equal spacing in both x and y directions). $F^{(i)}(x, y)$ are interpolated from the input image $E^{(i)}(x, y)$

$$F^{(i)}(x, y) = E^{(i)}(\bar{x}, y) + (x \pm \frac{fb}{2z} - \bar{x})[E^{(i)}(\bar{x} + 1, y) - E^{(i)}(\bar{x}, y)] \quad (3.3)$$

where

$$\bar{x} = \text{floor}(x \pm \frac{fb}{2z}). \quad (3.4)$$

The $\text{floor}(x)$ function returns the greatest integer towards minus infinity.

Matched grid is used to implement the cost function. In this implementation, p , q and z are chosen all to be the same size as the image arrays. In such a representation, all the functions are sampled on the same grid, thus the name *matched-grid representation*. This approach uses vertex-centered surface derivatives that are valid at each vertex of the z grid. Since p and q are the same size as z , some type of approximation must be made at the array edges. Bicubic interpolation is used to extrapolate the estimates.

3.2.3 Optimization

This algorithm uses direct optimization via the conjugate gradient method. This method has two advantages, the first is the guarantee reduction of the cost function at each step; the second is that no Hessian needs to be computed or stored.

3.2.4 Solution Techniques

The cost function is of the form

$$\min_z J = \int \int L(u, u', u'', \dots) dx dy \quad (3.5)$$

where u are the optimization variable, and L is a possible non-linear function of the optimization variables and its derivatives. The integral is taken over the domain of the variables. The solution to this problem can be found by solving the associated *Euler-Lagrange* equations (See a variational calculus book, such as [8]).

The Euler-Lagrange equations for a problem such as the one above are typically coupled non-linear equations. Such equations are usually very difficult to solve analytically but can sometimes be solved numerically by converting them into discrete equations. The conversion process involves substituting discrete approximations for any derivatives of the optimization variables. The optimization variables may have approximated by a discrete vector as well. The equations are then re-arranged to create iterative update of the form

$$u(k+1) = f(u(k), u(k-1), \dots) \quad (3.6)$$

where $u(k)$ is the value of the optimization variables for the k -th iteration.

When u has many components and when the components are updated in sequence based on the best current estimate u , the resulting update scheme is called a *Gauss-Seidel* optimization. When all of the components of u are update simultaneously based on a previous estimate for u , the resulting scheme is called a *Gauss-Jordan* optimization. *Gauss-Seidel* optimization schemes have higher convergence rates and are more robust, and are best implemented on a serial computer. *Gauss-Jordan* schemes, while they have lower convergence rates and are not as robust, can be implemented on parallel computers.

Another way of solving the optimization problem posed is by using direct optimization techniques. In this case the cost function, instead of *Euler-Lagrange* equations are discretized. Any integrations are approximated by sums and any derivatives are approximated by differences. The resulting cost function is of the form

$$\min_u J = \sum_x \sum_y f(u) \quad (3.7)$$

where $f(u)$ is a discrete approximation of $L(u, u', u'', \dots)$. We chose *conjugate gradient* optimization. The conjugate gradient scheme doesn't require the formation of the problem Hessian², which for an optimization problem with N variable is a N -by- N matrix. The conjugate gradient scheme is important for vision problem since for a typical 256-by-256 image, the Shape-from-Shading problem would have 256^2 or 65536 optimization variables. The hessian for this problem would have 256^4 or over 4 billion elements.

3.2.5 Speedup Techniques

For vision problems there are two promising speed up techniques: the use of *hierarchical basis functions*, and *multi-grid methods*. Both try to speed up the optimization problems by increasing the information transfer spatially. The methods are based on the properties of many vision algorithms where an optimization variables within a grid of optimization variables may only be affected by its nearest neighbors. Due to the local connectness, many vision algorithms have diffusion like properties, the solution must diffuse throughout the grid. Schemes that transfer information over longer distances thus may speed up an algorithm.

Using hierarchical basis functions transform the optimization space as seen by the optimization algorithm but not as seen by the vision algorithm. Basically it is like change of basis. Figure 3-4 shows the presentation of of a 9-by-9 domain in hierarchical basis. In particular, note that the nodes of the hierarchical basis propagate information over a much larger range than the nodes in nodal basis. It shows linear interpolation between nodes, but any interpolation scheme can be used to build a hierarchical basis. The variables can be transformed to the nodal basis when

²A linear approximation to the second order properties of the solution space at a given point

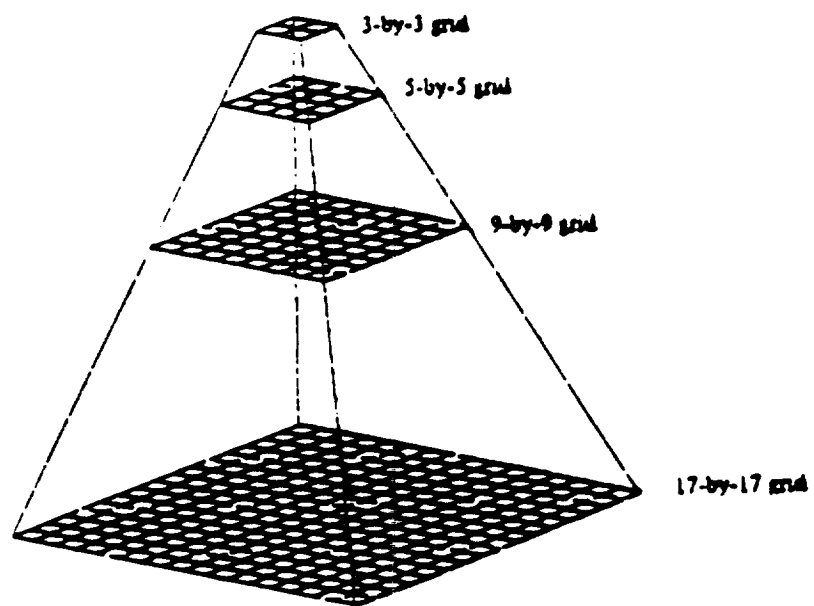


Figure 3-4: Hierarchical Basis Functions

computing the cost function or gradient. Unfortunately, all these transformation can introduce round-off errors may adversely affect sensitive algorithms.

The hierarchical basis functions have the most effect on the convergence and are the easiest to implement when the grid size is $2^n + 1$, where n is any positive integer. for such a grid it is possible to use $n + 1$ hierarchical basis levels. Using hierarchical basis functions increases the communication between nodes in the image array. so to speed up the diffusion process considerably.

The multigrid methods seek to propagate information over a large range by solving a series of problems of different size. Usually the original problem is formulated on grids that decrease in size by a factor of two when going from one layer to the next. The solutions on one layer are related to solutions on the layers above and below via interpolation or prolongation. The solution are kept consistent with each other via both intra-layer and inter-layer process. (See [23] and [2]).

Multigrid methods have the potential to be much faster than the hierarchical basis functions since most of the computation is done on the smaller layers. Multigrid methods are well suited to linear problems but may not work for non-linear problems. Since

$$\sum f(u) \neq f(\sum u) \quad (3.8)$$

for non-linear function f , and the multigrid methods rely on equality of the previous equation to constrain the solutions on the smaller grids so that they don't bias the solution on the larger grid.

One type of multi-grid method that can be used for non-linear problems is the *coarse-to-fine* method. In this method, the problem is solved on coarse layers first and the solution to each layer provides the initial condition to the next finer layer below. This method is significantly faster than just optimizing on the finest grid but does not produce as much convergence speed up as the full multigrid method.

Like the hierarchical methods, the multigrid methods work best when the grid size

is $2^n + 1$. However, since the multigrid methods define a series of problems rather than just choosing a new set of basis functions, the multigrid methods can be implemented easily for all grid size. There is some evidence, that the grid size reduction should be near 2 for best convergence rate (See [23]).

3.2.6 Discussion

The question of existence and uniqueness come up when working with optimization algorithms. For the cost function presented earlier, it is clear that a solution exist. The solution are bounded from below by zero. That is, the best possible value for the cost function is zero and can be achieved only when the estimated surface images and the actual images match exactly and when the regularization term is set to zero.

The uniqueness of a solution depends a great deal on the surface to be estimated. In general, both global and local minima will exist. The optimization techniques discussed above only guarantee convergence to a local minima. The global minimum may only be achieved if the initial conditions for the optimization algorithm are close to the true solution.

Chapter 4

Synthetic Image Test Results

It is crucial to test an new algorithm's performance, to understand its strength and weakness before applying it to the real images. To test, it must be possible to compare the estimated surface with the actual surface. The way to do it is to create synthetic images from a known surface topology, estimate the surface with *z-only* Depth from Shading and Stereo algorithm, compare the estimated surface with actual surface. Only after that, we can be confident in the correctness of the new algorithm.

4.1 Synthetic Images

Four synthetic images are used with various difficulties to the Depth from Shading and Stereo, from three typical topology.

- **Easy Crater Images:** The first pair of images is based on a crater on a flat plane. The light sources are oblique, this makes it relative easy for DFSS to estimate.
- **Hard Crater Images:** The second pair of images is based on a crater on a flat plane. The light sources are almost behind the camera, this makes it difficult for DFSS to estimate.

	b	f	z_0	$\Delta z/z_0$	$p_s^{(1)}$	$q_s^{(1)}$	$p_s^{(2)}$	$q_s^{(2)}$
Easy Crater	500	-2750	-997	0.0038	0.2	-0.5	-0.3	0.1
Hard Crater	500	-2750	-997	0.0038	0.1	0.1	-0.1	0.1
Hills	500	-12222	1000	0.0011	1.0	1.0	0.3	0.1
Mountain	100	-2292	-996	0.0153	0.5	0.5	-0.5	0.0

Table 4.1: Camera Geometry

- **Hill Images:** The third is based on a fractally-generated set of rolling hills. The light sources are oblique.
- **Mountain Images:** The third is based on a fractally-generated set of mountain terrain. The light sources are oblique and the baseline is smaller. This set of images poses a challenge and it most related to planet images.

The calibration parameters for the test images are summarized in Table 4.1. The table lists value for the baseline distance b , camera focal length f , nominal depth z_0 and light source vector $p_s^{(1)}$, $q_s^{(1)}$, $p_s^{(2)}$, $q_s^{(2)}$. Notice the focal length and nominal depth are negative so that it is consistent with a right hand system in perspective projection. Baseline b , depth z and light source components are in units of miles. The camera focal length and pixel spacing are based on camera units, millimeters.

All the test images are generate noise-free to test the best performance of the *z-only* DFSS algorithm. All the results are presented with history of smoothness weighting term λ , history of cost function, history of estimated error, and the mesh plot of depth z .

4.2 Algorithm Performance

In this section, I will show the performance of *z-only* of the above four set of images. All images are 65-by-65 pixel and using 6 levels of hierarchical basis, which is the largest number of levels that can be used.

4.2.1 Easy Crater Images

This set are based on a crater on a flat surface. It is very simple, and thus serves a good test bed. The very different light source positions give very different shadings to each image. As shown in Figure 4-1 The very strong shading information makes it easy for DFSS to estimate. Contour plots of the reflectance map clearly shows the effect of lighting. The larger separation of reflectance contours gradient space makes it easy to constrain the possible set of feasible gradient directions from brightness. Apparent *z-only* DFSS algorithm correctly estimated the easy crater surface. It is interesting to notice the small anomaly in the lower left corner of the surface. With the lighting condition, this anomaly has little effect on the estimated surface and the programs' convergence. *z-only* DFSS algorithm has very good performance, the correct surface topology is obtained in less than 150 iterations. The DFSS results is shown in Figure 4-4.

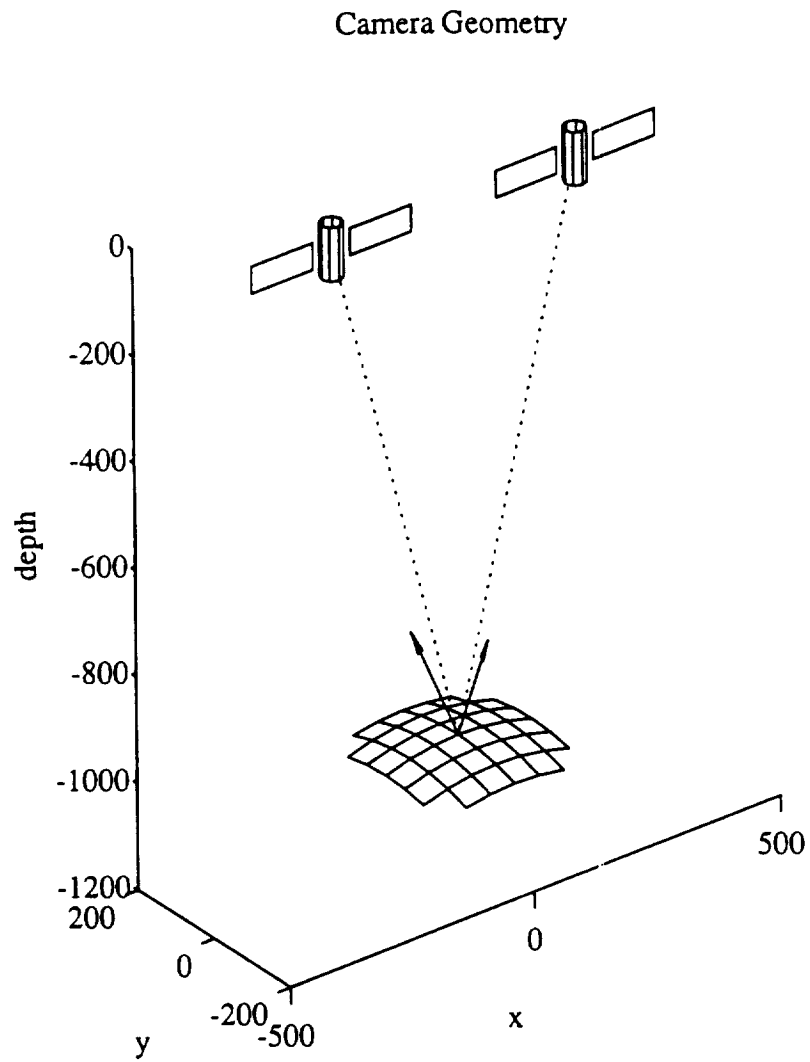


Figure 4-1: Easy Crater Synthetic Images Camera Geometry. Graph shows the camera position and direction(dotted lines) and light source direction(solid lines)

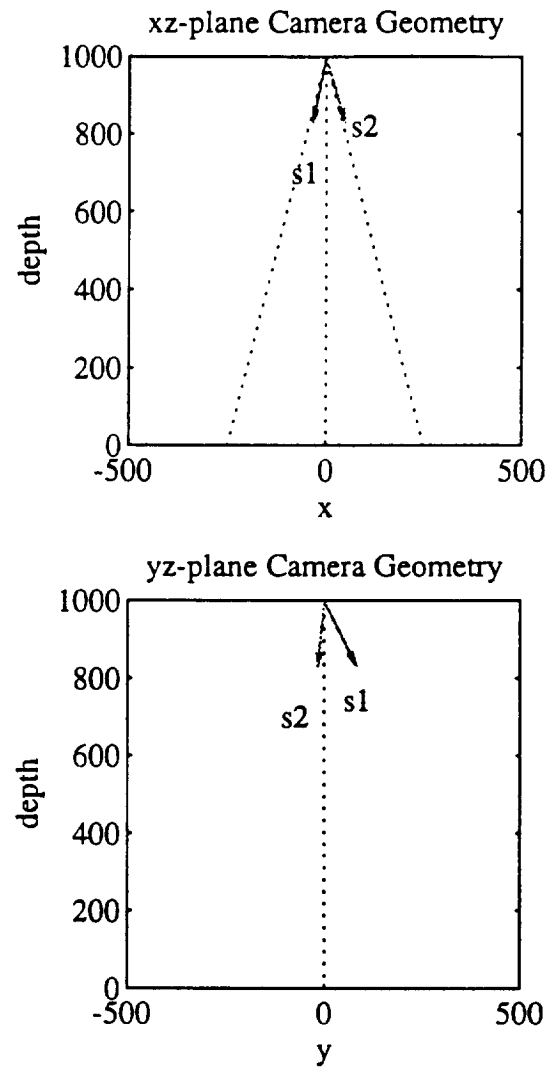


Figure 4-2: Easy Crater Synthetic Images Camera Geometry projected in xz and yz plane

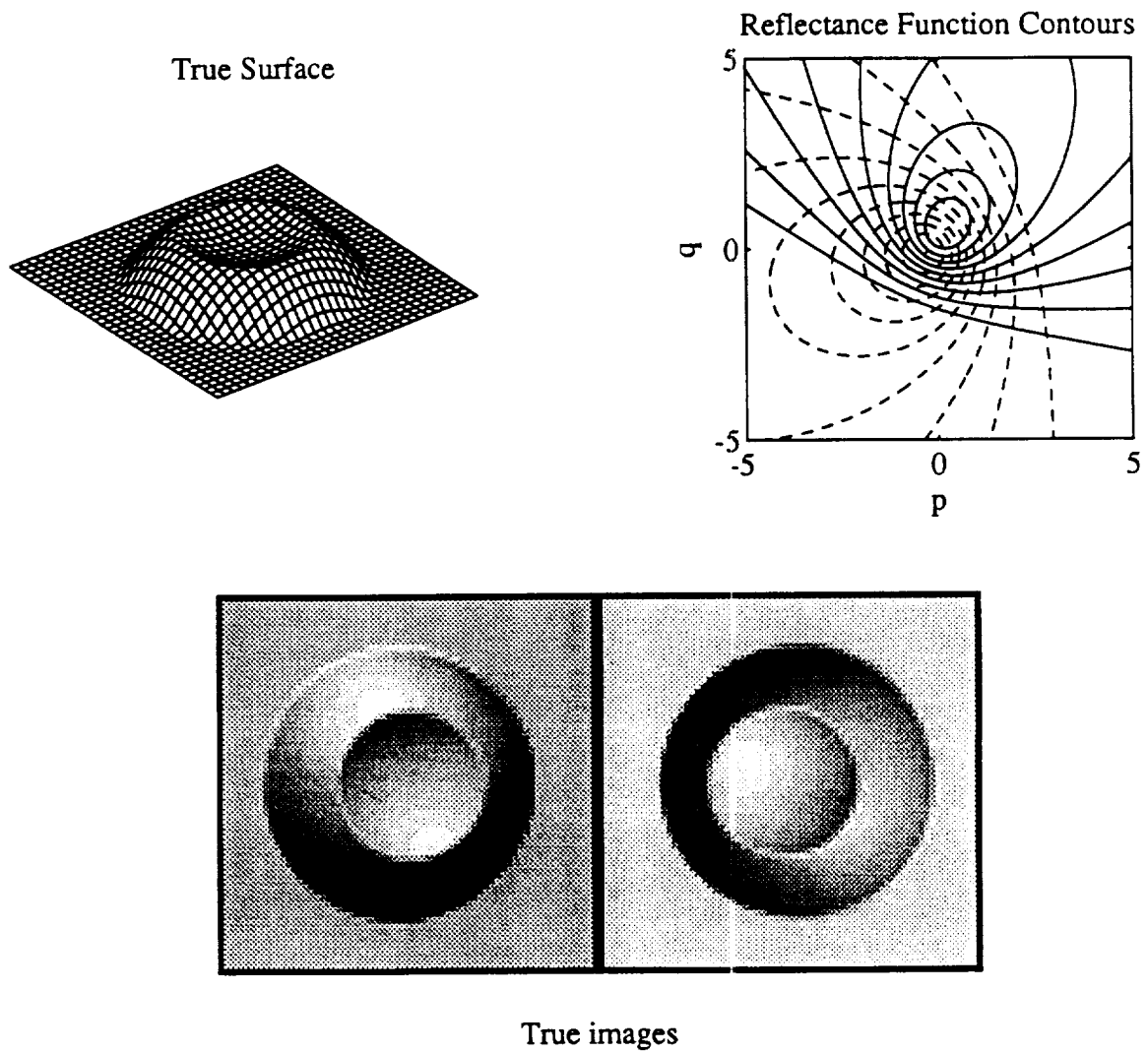


Figure 4-3: Easy Crater Synthetic Images

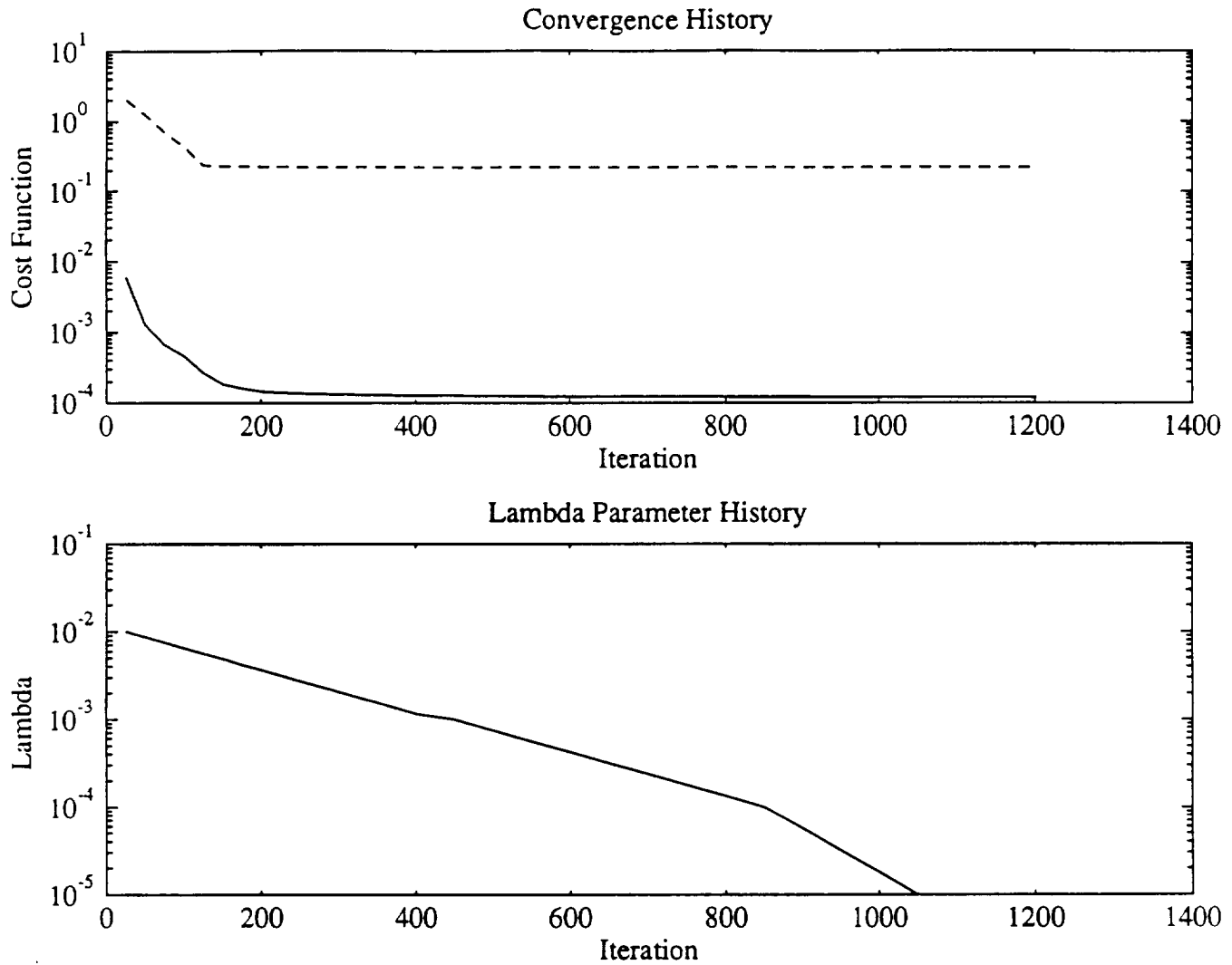


Figure 4-4: *z-only* DFSS iteration history on Easy Crater Images. Graph shows the *z-only* DFSS algorithm performance on the easy crater image pair. The upper graph shows the cost function(solid line) and RMS error(dashed line) of the estimated surface. The lower graph shows history of the smoothness weighting term λ

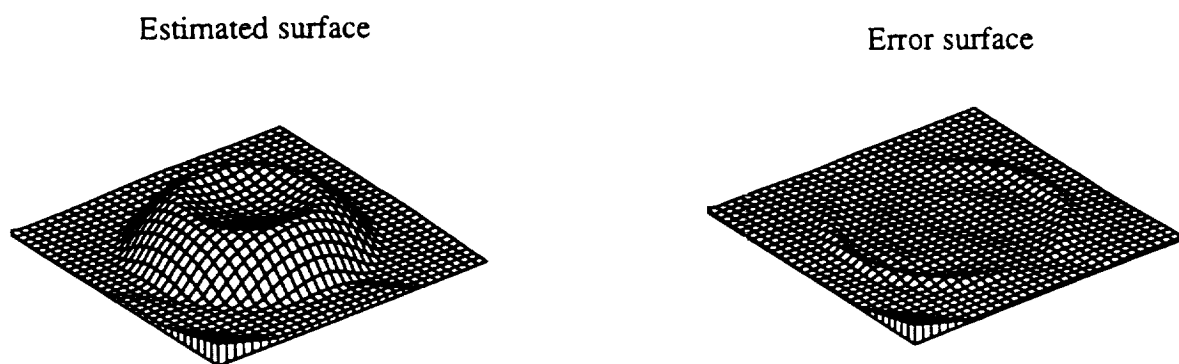
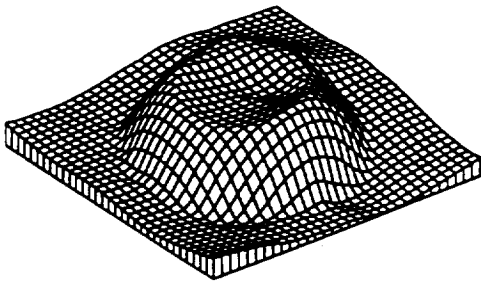
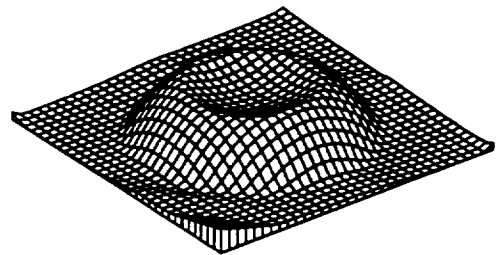


Figure 4-5: *z-only* DFSS Estimated Surface and Error on Easy Crater Images at the end of iterations

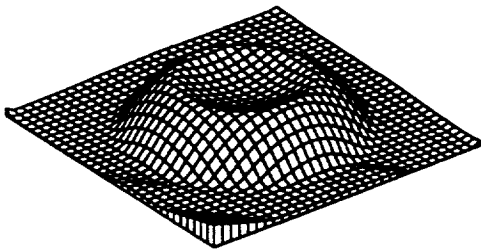
Estimated surface, iter. = 25



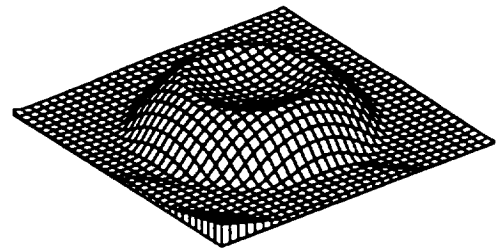
Estimated surface, iter. = 125



Estimated surface, iter. = 250



Estimated surface, iter. = 375

Figure 4-6: *z-only* DFSS estimation at various iteration steps on Easy Crater Images

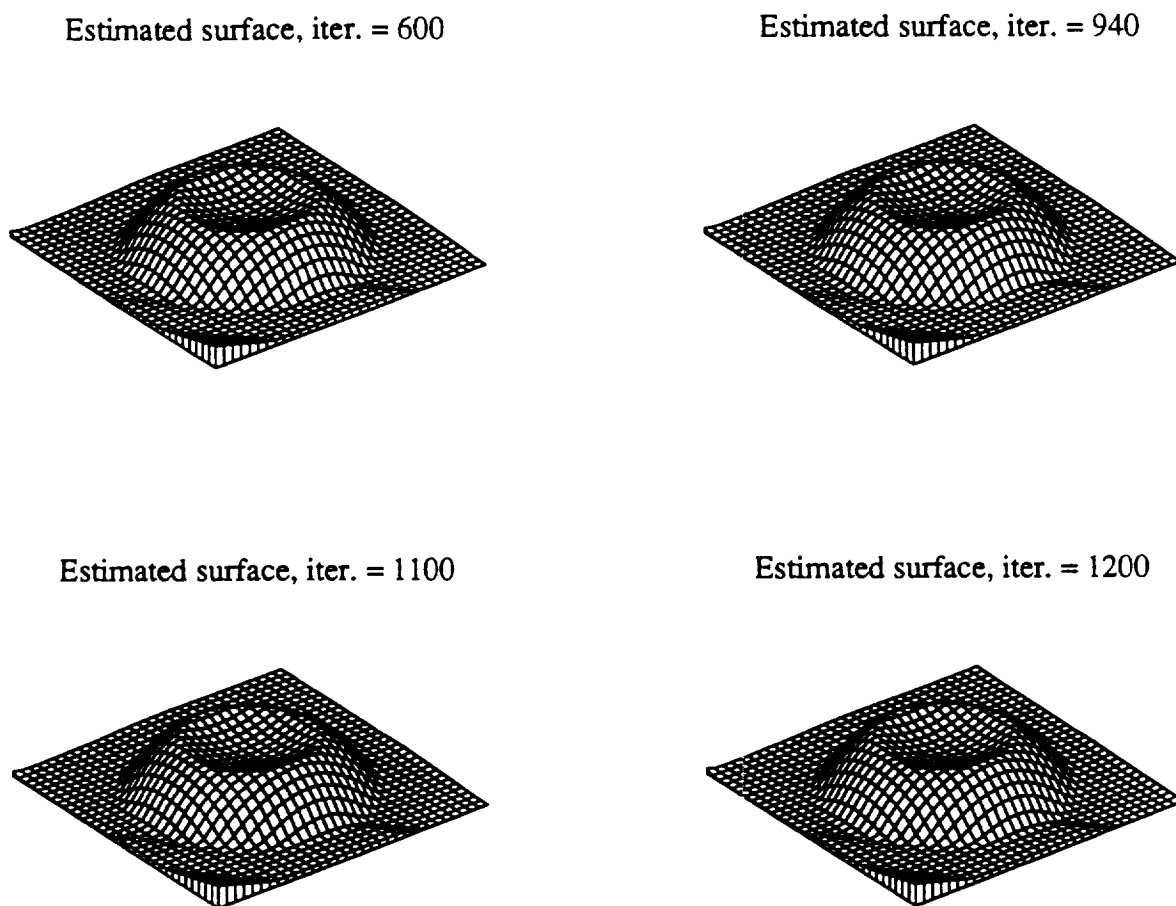


Figure 4-7: *z-only* DFSS estimation at various iteration steps on Easy Crater Images

4.2.2 Hard Crater Images

In this set of images, the baseline distance between the cameras is about half the distance to the surface and the light source positions for the two images are nearly the same and almost directly behind the cameras. As shown in Figure 4-8, from the light contour plot and the true images, the two images are very bright and look very much the same except for slight difference in the shading.

This images represent the worst case for DFSS algorithm since the shading information is weak and the range of brightness is small. However, they do have strong stereo correspondence information which is unfortunately not heavily utilized by *z-only* DFSS algorithm.

Figure 4-11 to Figure 4-14, shows the results of applying the *z-only* DFSS algorithm to this test case. It shows the estimated surface resembles the true image in figure 4-8, even though they don't match. Clearly the *z-only* DFSS algorithm gets stuck in a local minimum. The problem stems from that the algorithms incorrectly interpreted the surface to be concave when it is actually convex. Even though stereo information is presented and can be used to correctly determine the orientation of the surface, the *z-only* DFSS algorithm is biased towards shading information. The surface orientation ambiguity is also a result of having both light sources directly behind the cameras.

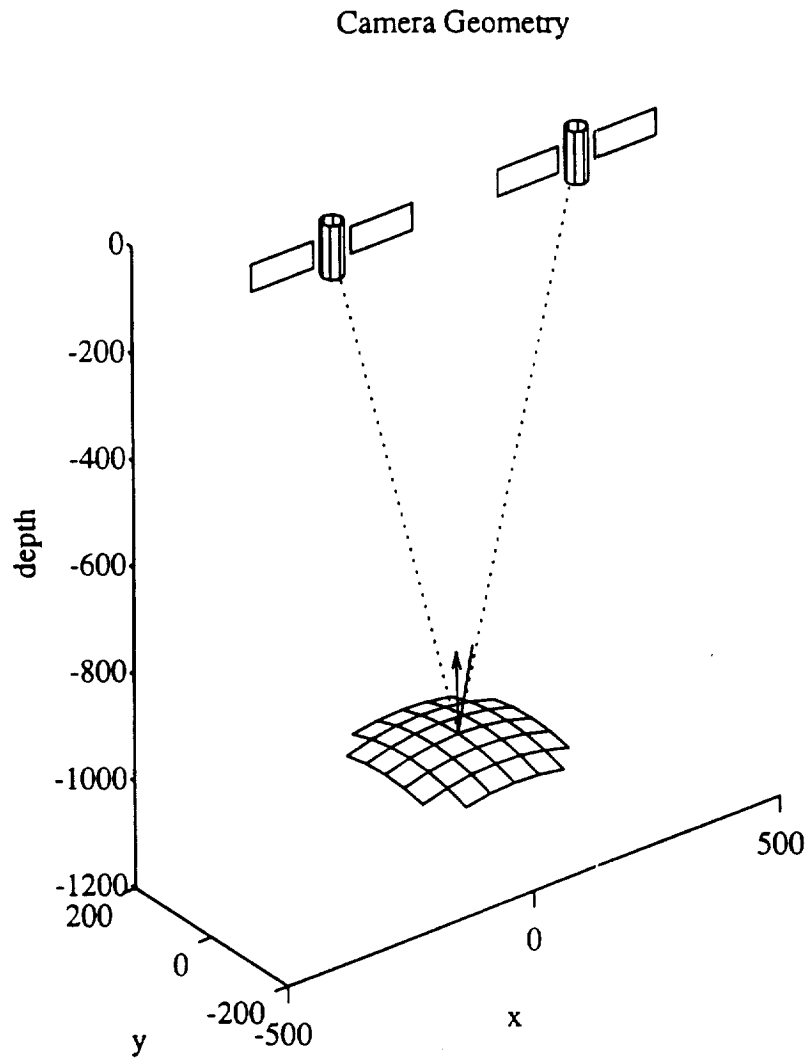


Figure 4-8: Hard Crater Synthetic Images Camera Geometry. Graph shows the camera position and direction(dotted lines) and light source direction(solid lines).

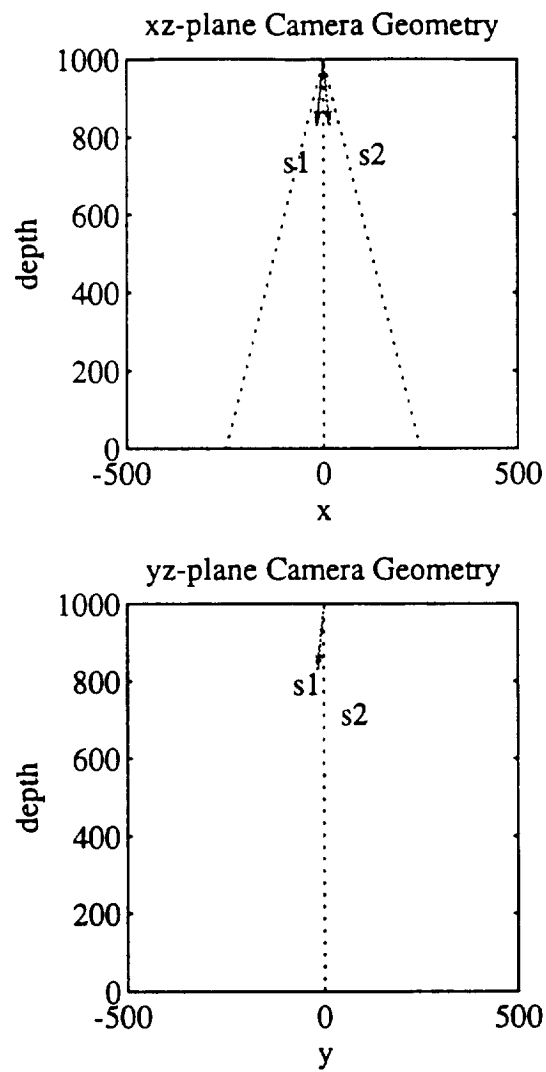


Figure 4-9: Hard Crater Synthetic Images Camera Geometry projected in xz and yz plane

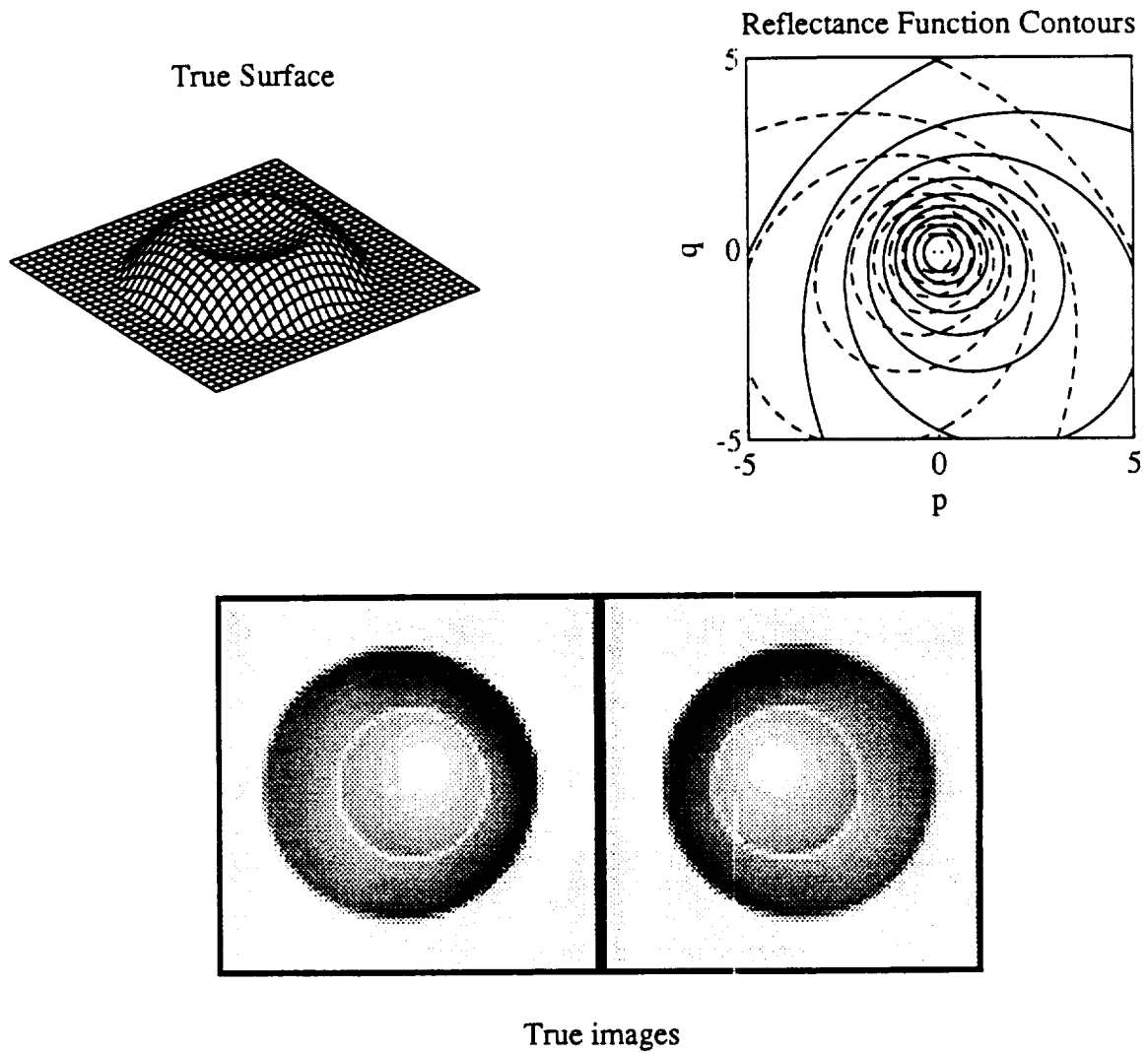


Figure 4-10: Hard Crater Synthetic Images

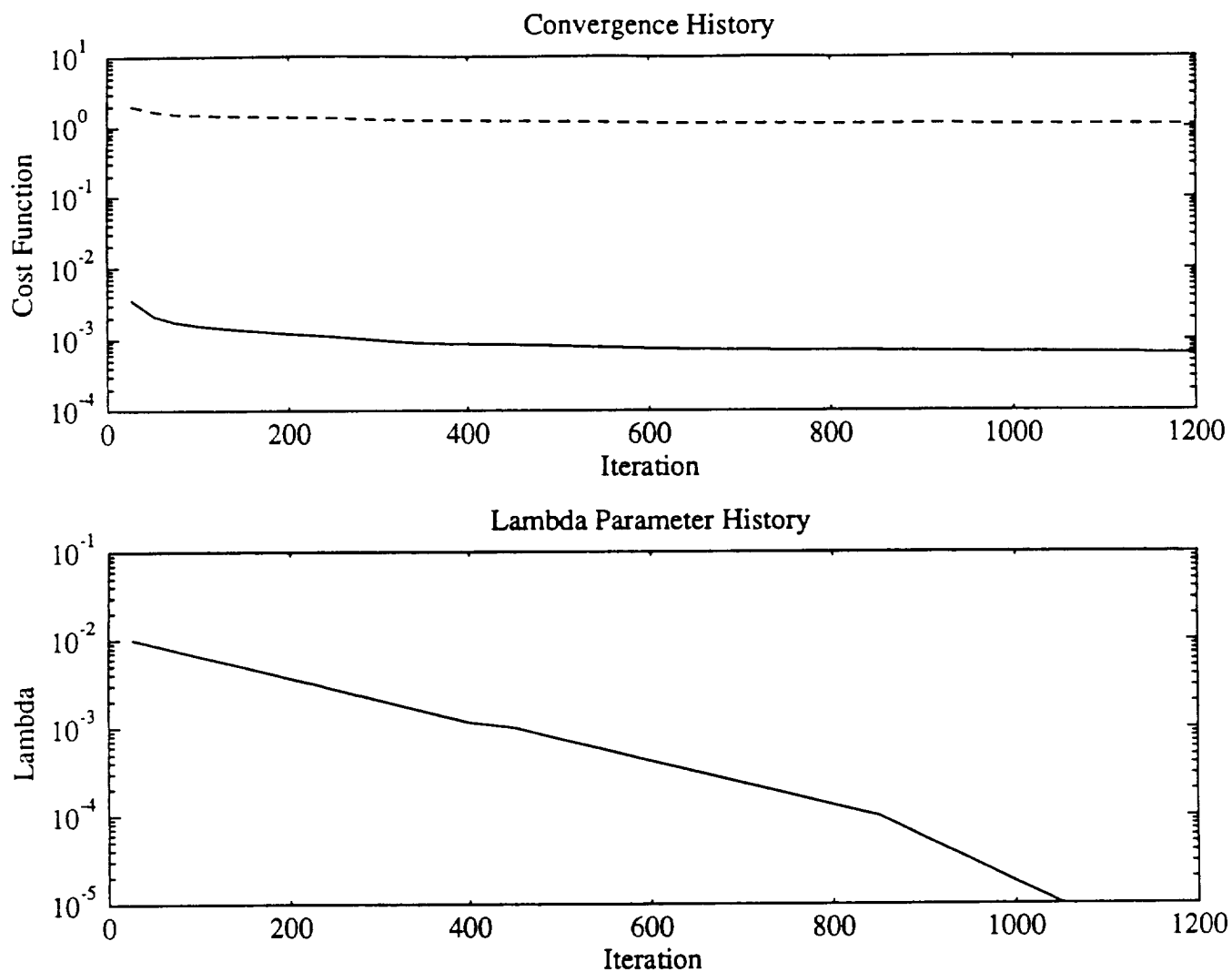


Figure 4-11: *z-only* DFSS iteration history on Hard Crater Images. Graph shows the *z-only* DFSS algorithm performance on the hard crater image pair. The upper graph shows the cost function(solid line) and RMS error(dashed line) of the estimated surface. The lower graph shows history of the smoothness weighting term λ

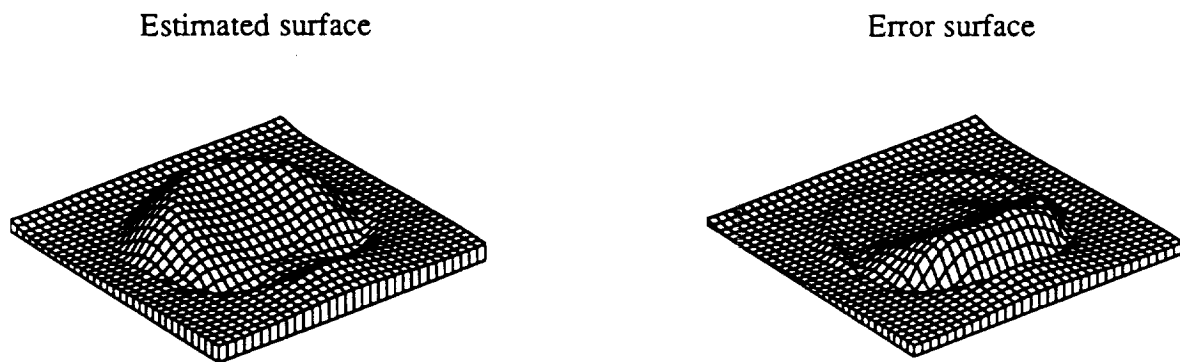
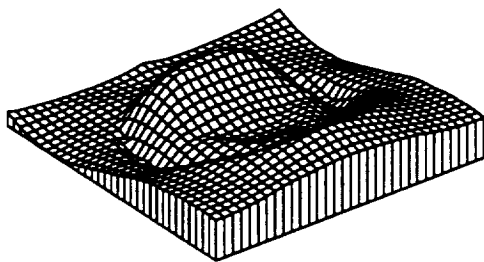
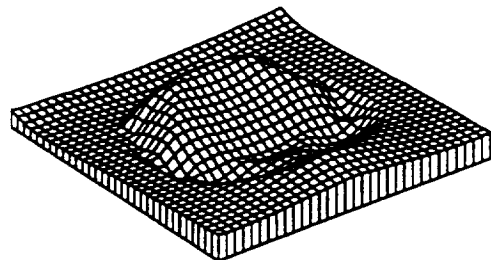


Figure 4-12: *z-only* DFSS Estimated Surface and Error on Hard Crater Images at the end of iterations

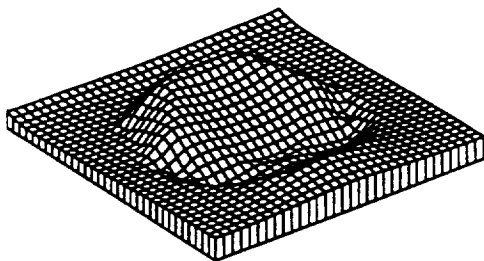
Estimated surface, iter. = 25



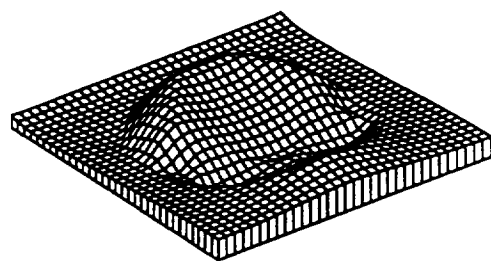
Estimated surface, iter. = 125



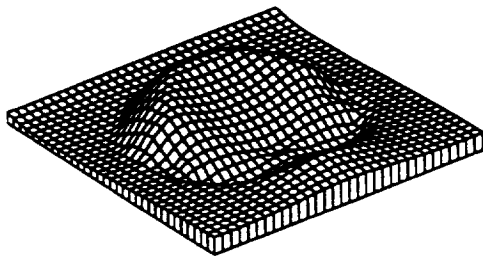
Estimated surface, iter. = 250



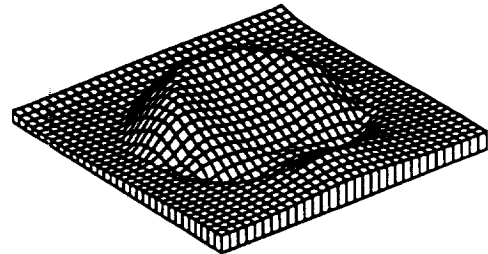
Estimated surface, iter. = 375

Figure 4-13: *z-only* DFSS estimation at various iteration steps on Hard Crater Images

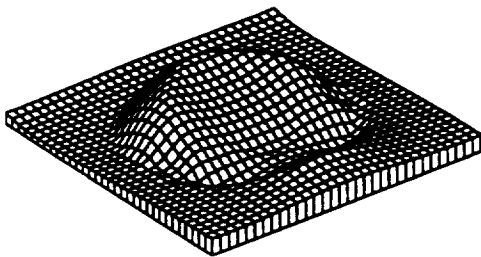
Estimated surface, iter. = 600



Estimated surface, iter. = 940



Estimated surface, iter. = 1100



Estimated surface, iter. = 1200

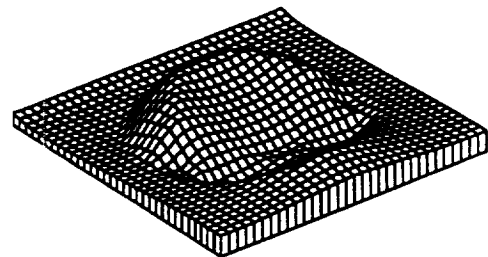


Figure 4-14: *z-only* DFSS estimation at various iteration steps on Hard Crater Images

4.2.3 Hill Images

The third set of images is of an undulating surface similar to erode hills and was generated using a fractal technique. Figure 4-15 shows the camera geometry, true surface, reflectance contours. This set of images is more representative of the type of terrain the DFSS algorithm are likely to encounter. As shown in Figure 4-15, the left cameras is directly over the surface and the light camera views the surface obliquely. The light sources are separated, as in the ease crater case. This results in the images with strong shading information. The DFSS algorithm performed rather well in this case, correctly interpreted the surface.

The Figure 4-18 shows the results of applying the DFSS algorithm to the hill images. *z-only* DFSS algorithm performs well, correctly estimated the complicated topology. Due to the complexity, the images requires more iterations to obtain a satisfactory estimate of the surface than the easy crater case.

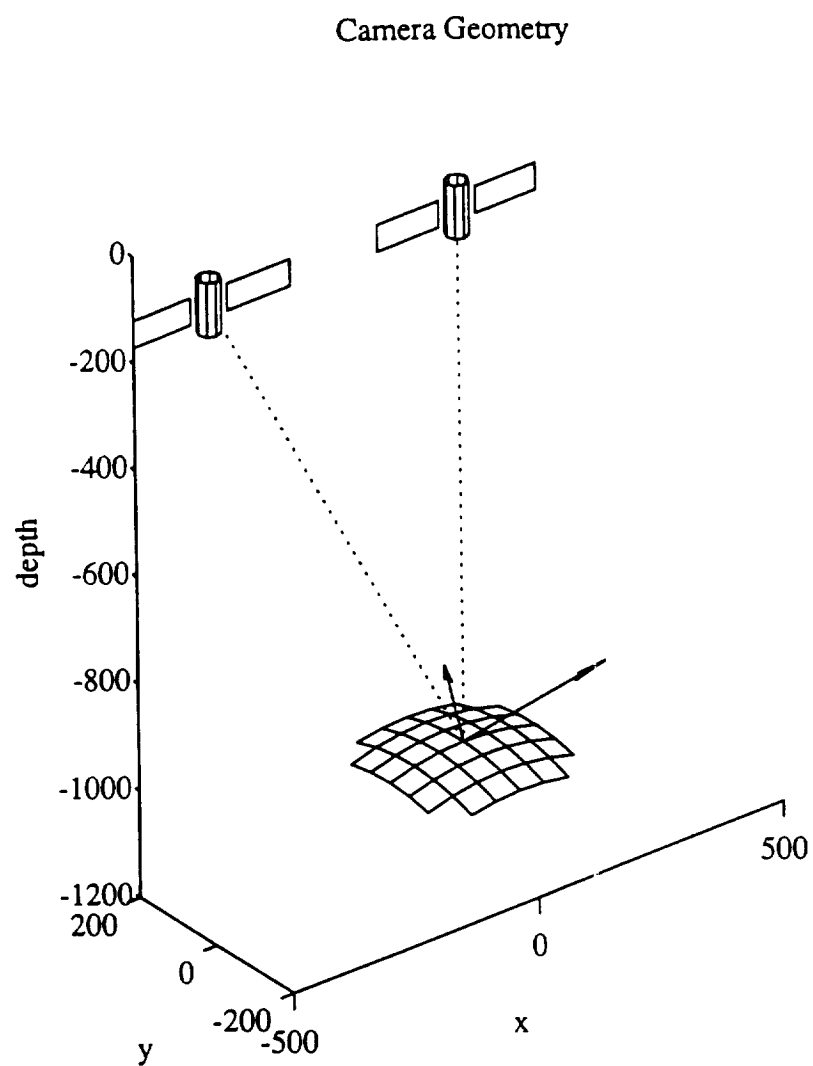


Figure 4-15: Hill Synthetic Images Camera Geometry. Graph shows the camera position, direction(dotted lines and light source direction(solid lines).

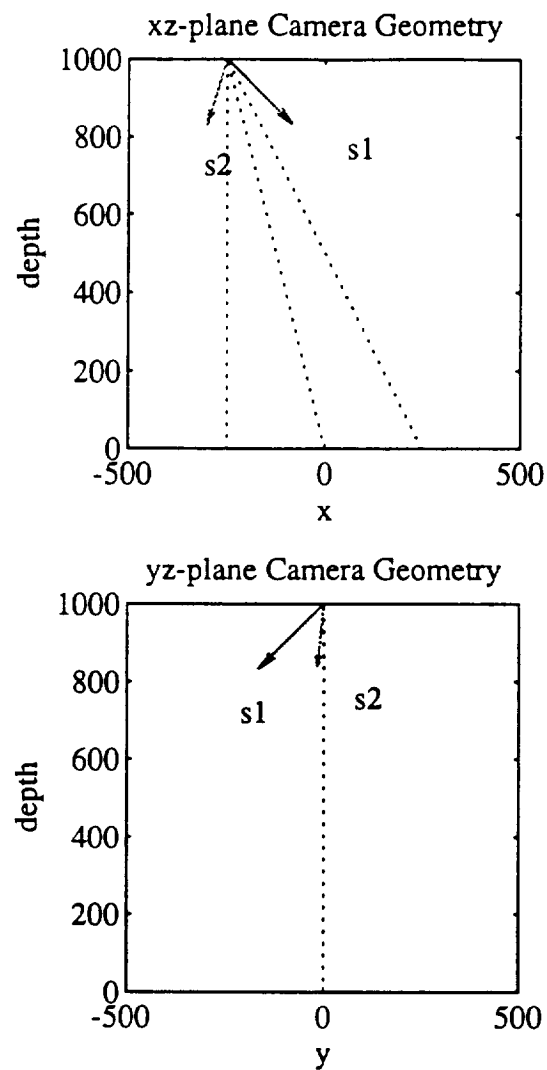


Figure 4-16: Hill Synthetic Images Camera Geometry projected in xz and yz plane

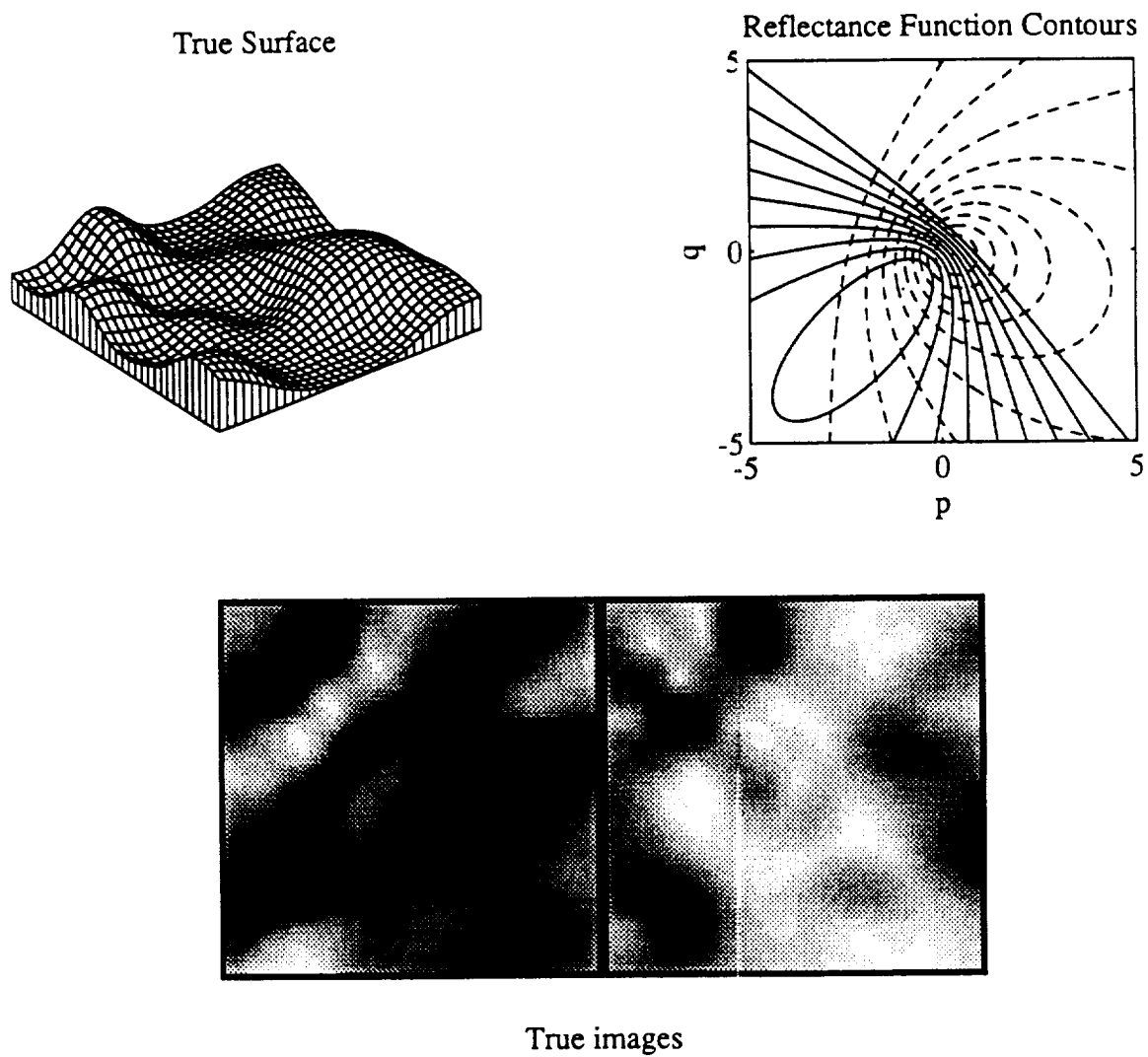


Figure 4-17: Hill Synthetic Images

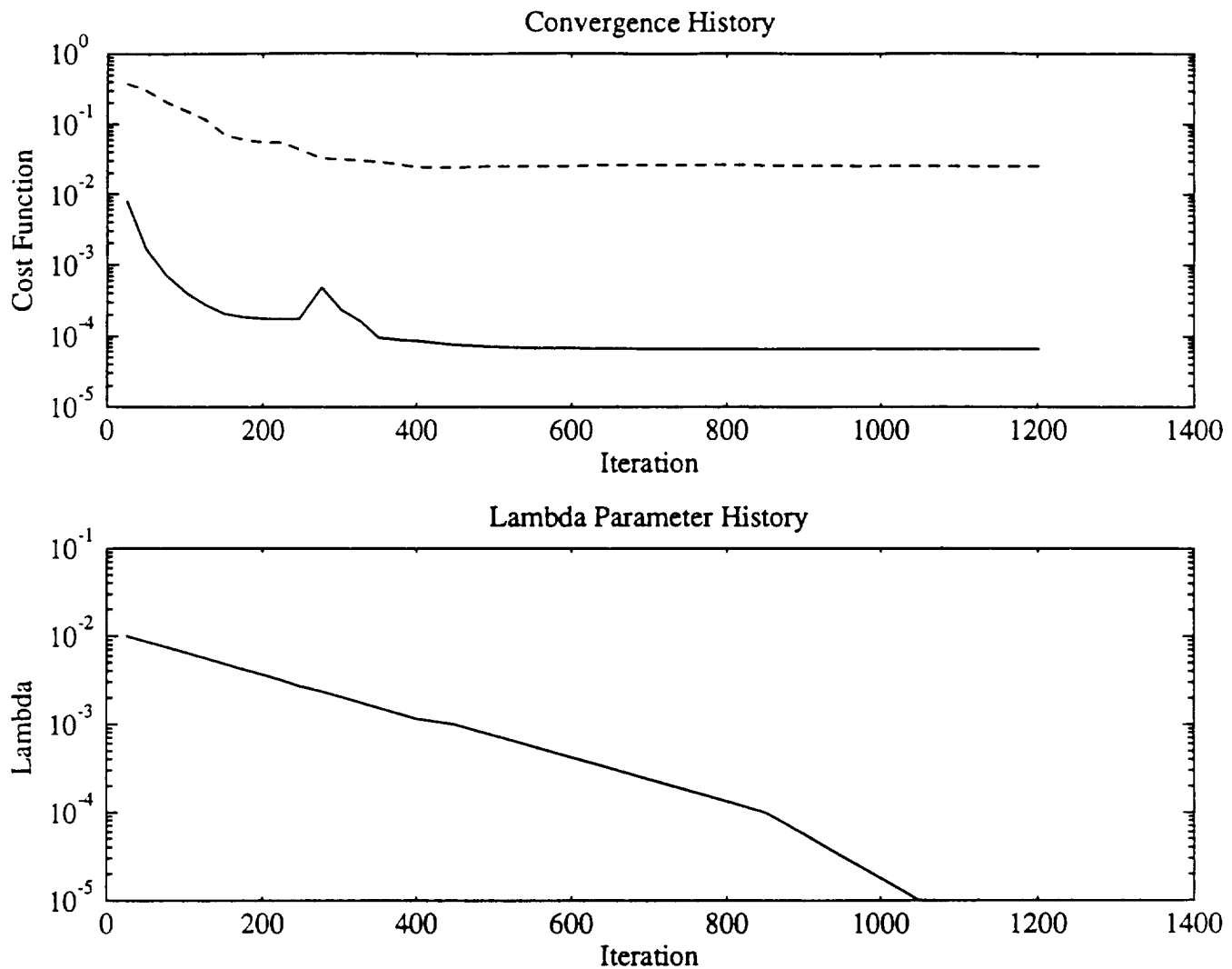


Figure 4-18: *z-only* DFSS iteration history on Hill Images. Graph shows the *z-only* DFSS algorithm performance on the hill image pair. The upper graph shows the cost function(solid line) and RMS error(dashed line) of the estimated surface. The lower graph shows history of the smoothness weighting term λ

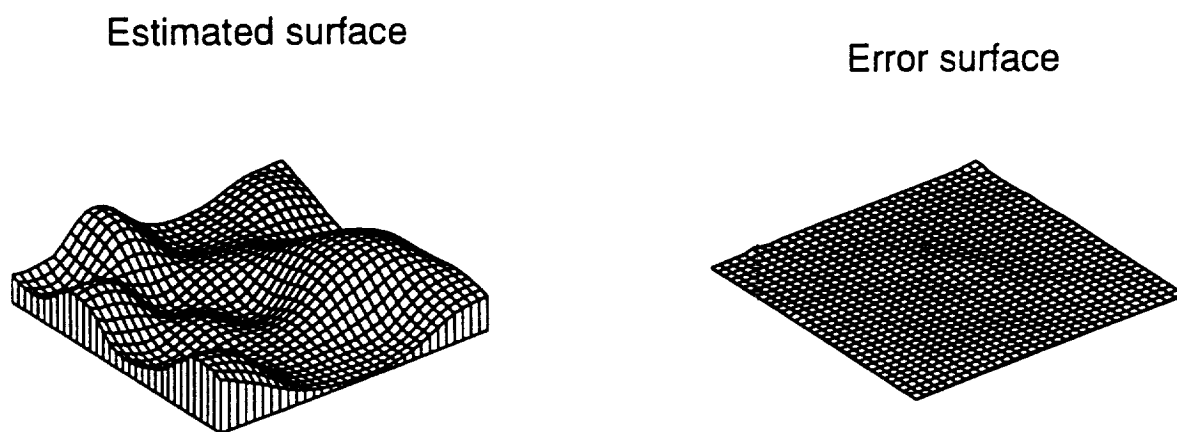
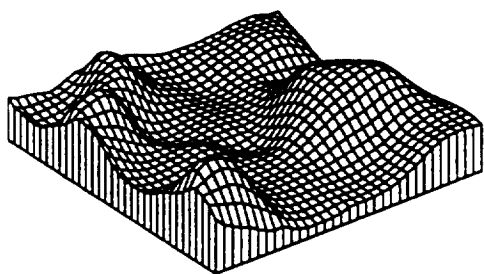
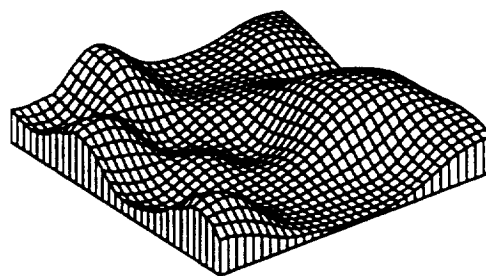


Figure 4-19: *z-only* DFSS Estimated Surface and Error on Hill Images at the end of iterations

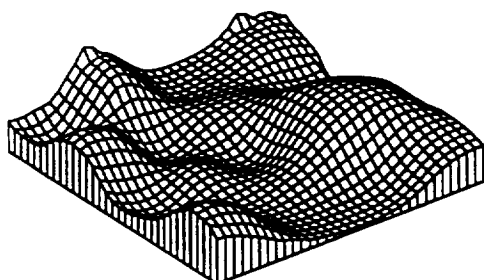
Estimated surface, iter. = 25



Estimated surface, iter. = 125



Estimated surface, iter. = 250



Estimated surface, iter. = 375

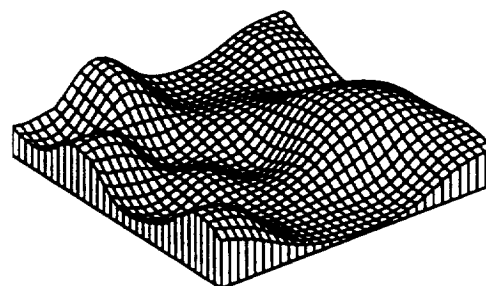
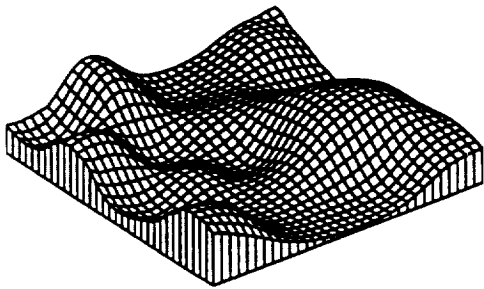
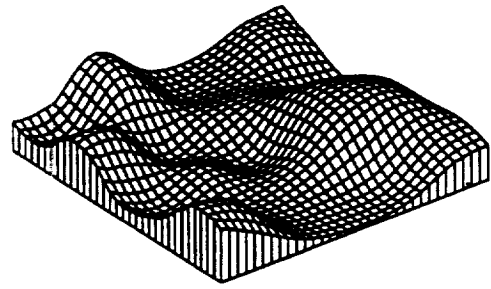


Figure 4-20: *z-only* DFSS estimation at various iteration steps on Hill Images

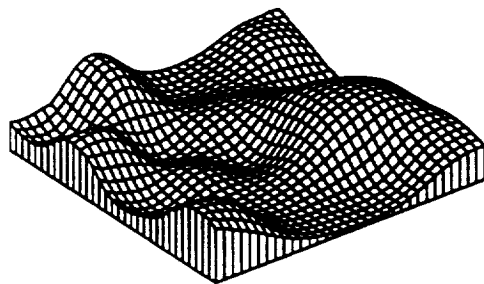
Estimated surface, iter. = 600



Estimated surface, iter. = 940



Estimated surface, iter. = 1100



Estimated surface, iter. = 1200

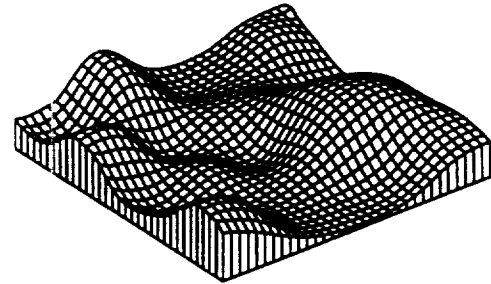


Figure 4-21: *z-only* DFSS estimation at various iteration steps on Hill Images

4.2.4 Mountain Images

The fourth set of images is of a highly mountainous surface and was created to show the performance on steep terrain. The steep terrain requires that the baseline distance from the cameras to be small so that all the surface points are visible from both cameras. Thus it is a good example to show the merit of the algorithm when the stereo baseline is small. As shown in Figure 4-22, The light source position are widely separated and generate deep shadows on this steep terrain. The reflectance maps are flat within a shadow so no helpful gradient is available to the algorithm. In addition, knowledge that a particular pixel is in shadow only constrains the set of possible gradient directions to a sub-plane of gradient space. Thus, within a shadow region, much more influence is given to the brightness values from the other image. This results in a slower convergence.

The Figure 4-25 shows the results of applying DFSS algorithm. The *z-only* DFSS algorithm performed reasonably well, correctly interpreted the surface. Due to the complexity of the surface. the algorithm requires many more iterations to achieve a satisfactory solution. Never the less, the algorithm correctly estimates the surface. It proves the algorithm is robust and can handle small base line difference.

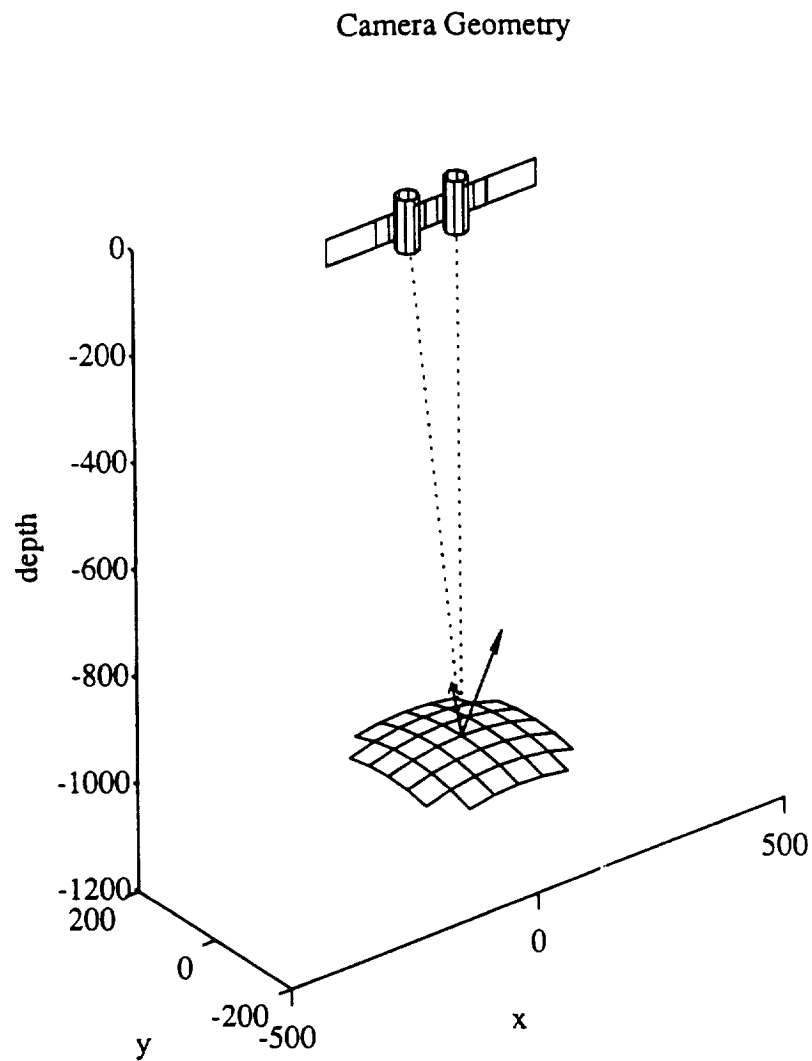


Figure 4-22: Mountain Synthetic Images Camera Geometry. Graph shows the camera position and direction(dotted lines) and light source direction(solid lines).

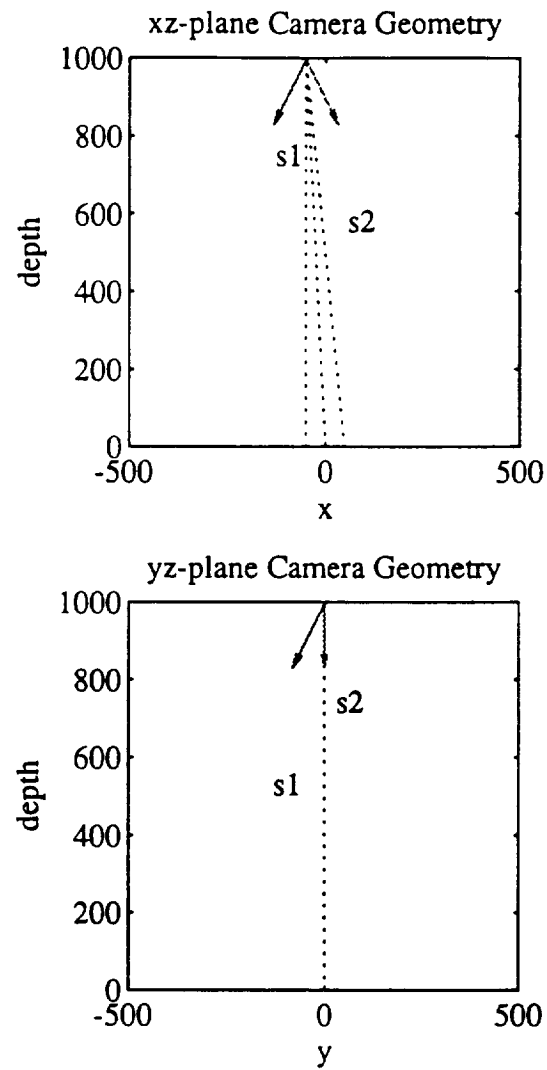


Figure 4-23: Mountain Synthetic Images Camera Geometry projected in xz and yz plane

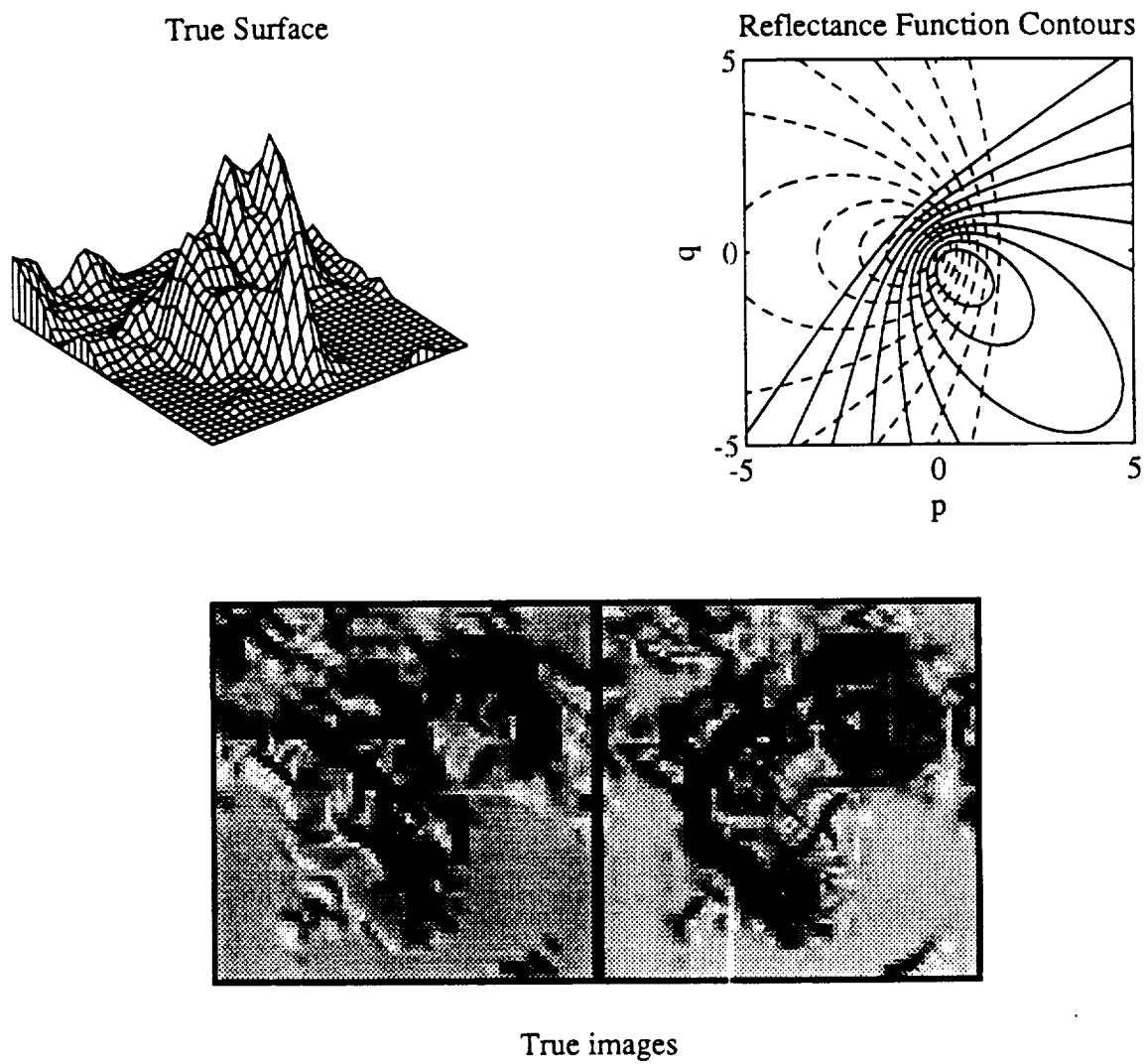


Figure 4-24: Mountain Synthetic Images

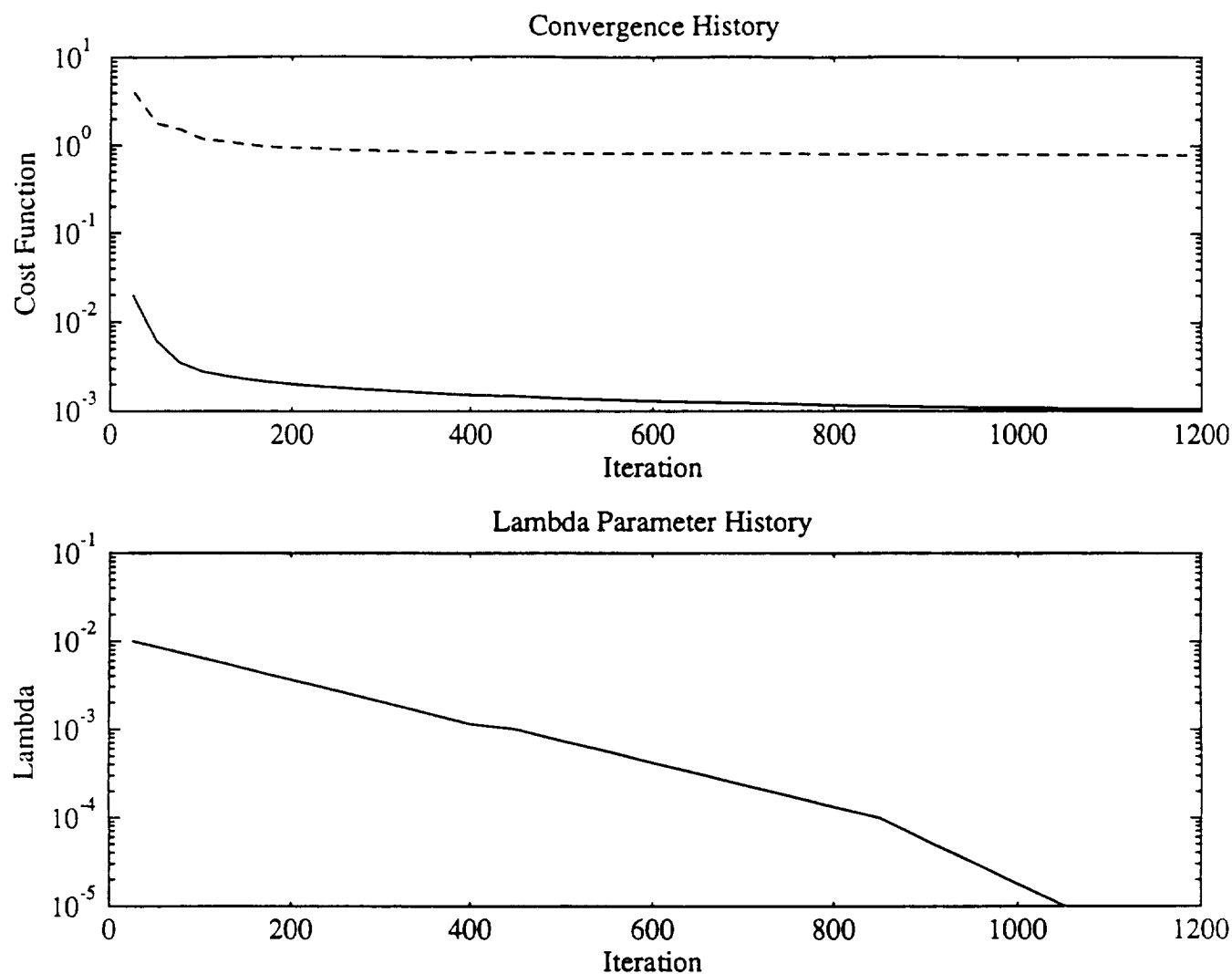


Figure 4-25: *z-only* DFSS iteration history on Mountain Images. Graph shows the *z-only* DFSS algorithm performance on the mountain image pair. The upper graph shows the cost function(solid line) and RMS error(dashed line) of the estimated surface. The lower graph shows history of the constraint λ

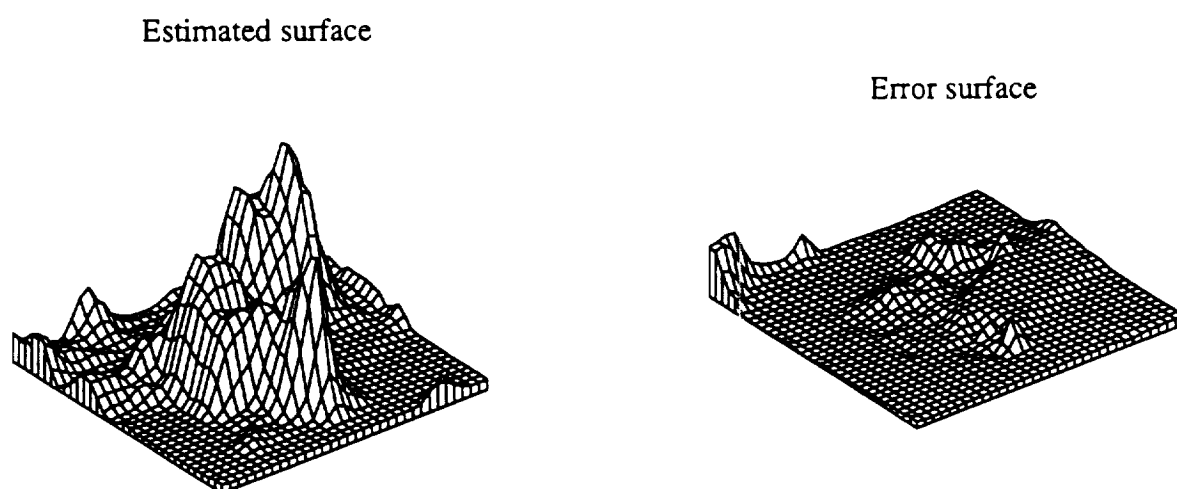
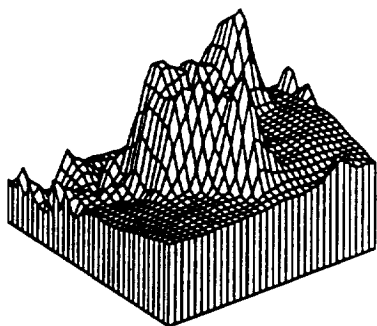
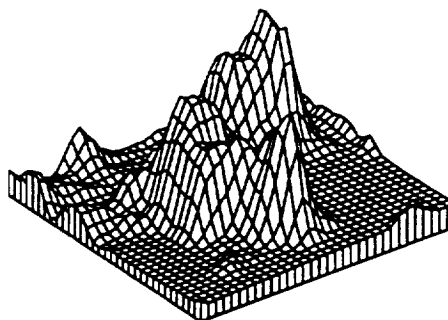


Figure 4-26: DFSS Estimated Surface and Error on Mountain Images at the end of iterations

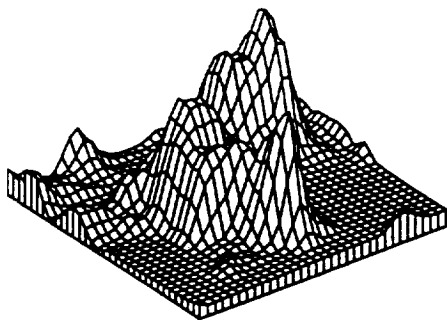
Estimated surface, iter. = 25



Estimated surface, iter. = 125



Estimated surface, iter. = 250



Estimated surface, iter. = 375

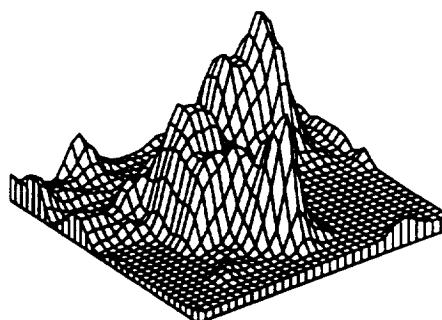
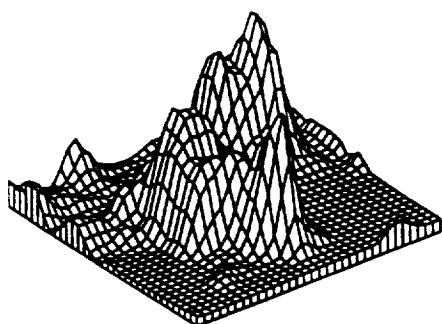
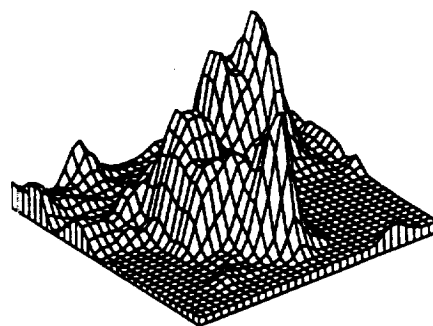


Figure 4-27: DFSS estimation at various iteration steps on Mountain Images

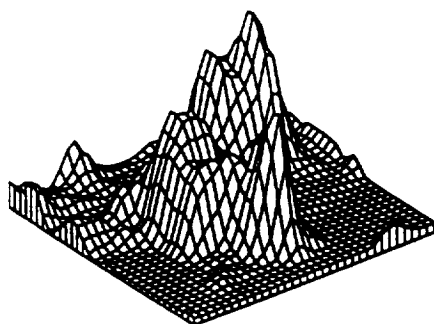
Estimated surface, iter. = 600



Estimated surface, iter. = 940



Estimated surface, iter. = 1100



Estimated surface, iter. = 1200

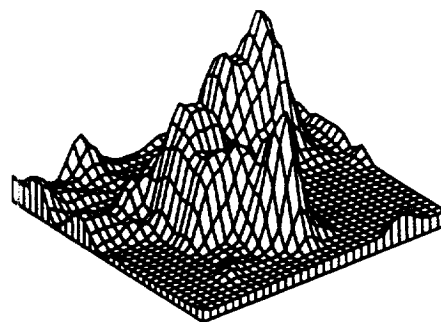


Figure 4-28: DFSS estimation at various iteration steps on Mountain Images

	Rel. Error	Abs. Error	Iterations
Easy Crater	0.172	0.173	1200
Hard Crater	1.072	1.073	1200
Hills	0.018	0.025	1200
Mountain	0.718	0.780	1200

Table 4.2: Performance Comparison

4.3 Summary of the Results

A summary of the results of DFSS algorithms running on easy crater, hard crater, hill and mountain images are presented in Table 4.2. The relative and absolute error between the true and estimated surface at the last iteration are calculated as

$$J_{abs} = \sqrt{\frac{1}{NM} \sum_{x,y} (z - z_{true})^2}, \quad (4.1)$$

and

$$J_{rel} = \sqrt{\frac{1}{NM} \sum_{x,y} ((z - \bar{z}) - (z_{true} - \bar{z}_{true}))^2}. \quad (4.2)$$

where \bar{z} and \bar{z}_{true} are the average of z and z_{true} respectively.

4.4 Performance

From the test results on the synthetic easy crater, hard crater, hill and mountain images, clearly *z-only* DFSS can achieve very good estimation on surface topology in a reasonable number of iterations, in the easy crater, hill and mountain image cases. *z-only* algorithms also proves to be robust, especially in small baseline situation. The only difficulty is dealing with images which has very weak shading information, as represented by the hard crater case.

It was shown that the *z-only* DFSS algorithm has much better performance using

the two images that a simple shape from shading algorithm which uses only one image.(see [24]) This performance increase validates the fusion approach to obtaining better vision algorithm.

The *z-only* algorithm was also shown to be robust, able to accurately estimate the synthetic surface in the presence of several types of errors. The performance of the algorithm based on images that contains noise, geometry error, or reflectance errors was shown in [24] chapter 7. In Most case, The algorithm was able to form a close estimate.

It is clear the algorithm has considerable difficulties with the hard crater images. The optimization got caught in the local minimum. The reason is clearly shown in the Figure 4-8 for a given brightness level(i.e., along one of the contours), there are two viable solutions with different surface orientations. The local minimum has a orientation in the “dipped” region that is viable but incorrect orientation, and the stereo information is not strong enough to pull it out of the local minimum.

The C version of *z-only* DFSS algorithm runs 2-3 times faster that its counterpart in Matlab, memory requirement is 30 percent less. It is also made more portable. The C functions has been tested on Sun, and IBM compatible PCs.

Chapter 5

Viking Image Results

In the previous chapter, the *z-only* Depth from Shading and Stereo algorithm has been shown to be efficient and robust. In this chapter, the algorithm will be applied to real images, namely areal photos taken during Viking Space Project.

5.1 Viking Space Project

Viking Space Project was started by the National Aeronautics and Space Administration on November 15, 1968. The main objectives of the project was to achieve a soft landing on the surface of the Mars and to relay scientific data back to the Earth. The main function of the orbital cameras, whose pictures are displayed in this thesis were to aid in the selection of safe landing sites to establish the geologic and dynamic environments in which the lander experiments were performed.

Two previous missions, using Mariner 4, mariner 6 and 7 obtained a blend of low-resolution, wide area pictures. Mariner 9, the final predecessor to Viking was the first spacecraft to go into orbit about another planet. It took more than 7300 images of Mars, covered the entire surface at a resolution of 1-3 km, and selected area down to 100 meters.

5.1.1 Viking Mission

The Viking Mission consisted of four spacecrafts: two identical orbiters and two identical landers. One of the orbiter experiment was the Visual Imaging Subsystem (VIS), which acquired the images that are used in this thesis. The major objective of the VIS experiment was to characterize potential landing site.

Viking Orbiter I was launched from Kennedy space center at Cape Canaveral on August 20, 1975, and arrived at Mars on June 19, 1976. Initially, the spacecraft was put into a Mars-synchronous elliptical orbit with a period of 24.66 hours, an apsapsis of 33,000 km and a periapsis of 1513 km. During the first month, Viking I began systematically imaging Mars on July 20, 1976. Its highly elliptical orbit was particularly suited for studying the surface because it allowed a mix of close-up, detailed views at periapsis and long range synoptic view near or at apoapsis. More than 30,000 pictures were taken.

Viking Orbiter II was launched September 9, 1975, and arrived at Mars on August 7, 1976. A major difference in the orbit of this spacecraft compared to that of Viking Orbiter I is its high inclination, which allowed Viking Orbiter II to observe the complex, enigmatic polar regions at relative close range. Viking Orbiter II returned nearly 16,000 pictures of Mars and its satellites.

The Viking orbiter spacecrafts operated in orbit around Mars from 1976 to 1980. The orbiter imaging systems imaged all of the terrains on Mars, collected some color and stereo images, and made observations of phobos and deimos. Some image sequences acquired by the VIS experiment include systematic medium and high resolution coverage of large portions of the surface, stereo images, observations of Phobos and Demios, color images of the equatorial regions, observations of the polar regions, and monitoring dust storm activity.

5.1.2 Viking Orbiter Visual Subsystem

Each Viking Orbiter, Viking Orbiter I and II, was equipped with two identical vidicon cameras, called the Visual Imaging Subsystem (VIS) [26], [14], [1]. Each VIS camera consisted of a telescope, a slow scan vidicon, a filter wheel, and associated electronics. The angular field of view of the camera as defined by the reseau pattern was 1.51 by 1.69 degrees. The ground area covered by an image varies as the function of spacecraft altitude and emission angle. A digital image was generated by scanning the vidicon face plate. A full resolution, uncompressed Viking orbiter image consists of an array of 1056 lines with 1204 samples per line. There are only 1182 samples in each line are valid. The extra 22 are consist of dark bands on the left and right edge of each image, produced by an opaque mark in front of the camera. The images then transmitted back to earth station.

Many Viking orbiter images have missing data and contain some amount of noise [14]. The missing data are mainly due to sample intervals, resulted from the raw data being stored on the spacecraft and transmitted to earth in packets that contain every seventh pixel. The noise found in these images include single-pixel random noise and several source of coherent noise. The random noise is usually due to telemetry errors. Techniques exist to remove the random noise and missing data [4]. The coherent noise arises from shuttering of the adjacent camera, filter wheel stepping, and scan platform movements [14]

5.2 Data selection

Mars provides many features are suitable to use DFSS algorithms to analysis the photo images. Great canyons are incised into the surface, huge dry river beds show the changes in topology features, large volcano tower distinct itself on a flat surface.

However, huge volume of Mars images also pose difficulties for image selection along with some natural obstacles, such as seasonal dust storm, caps od carbon dioide

at the poles.

Image pair selection are done by Mike Caplinger. Using their local facilities, Mike Caplinger is able to browse through a large amount of Mars images and pick out the images of the same longitude and latitude. The associated Sun position and space craft position and angles are obtained. These geometry information and the image data are then sent to us to use with DFSS algorithm.

5.3 Data Processing

Two preliminary processing steps are always done to radiometrically and geometrically calibrate the images:

- Viking Orbiter images are radiometrically calibrated by converting the digitized signal from the camera into a quantity that is proportional to the radiance reaching the sensor. Each Viking camera was calibrated before flight. In addition, changes in the calibration over time has been estimated from analyses of images of deep space and dust storms. The radiometric calibration procedure applies additive and multiplicative corrections that account for the varying sensitivity of the vidicon across the field of view and over time. The calibrated values are proportional to radiance factor, which is defined as the ratio of the observed radiance to the radiance of a normally illuminated Lambertian reflector of unit reflectance at the same heliocentric distance.
- Geometric calibration of Viking Orbiter images removes electronic distortions and transforms the point perspective geometry of the original image into a map projection. The electronic distortions are barrel-shaped distortions from the electron beam readout and complex distortions from interactions between the charge on the vidicon face plate and the electron beam. The electronic distortions are modeled by comparing the predicted locations of undistorted reseau marks with the actual locations in an image.

During the processing, resseau marks are removed, bit errors and/or tape errors are corrected. The raw Mars images are projected using the nominal imaging geometry to a projection such as sinusoidal or Mercator, relative to a reference spheroid. This has the effect of removing any gross effects caused by the curvature of the planet, but since the small scale topography is not modeled, the disparities it induces remain. The images are also rotated so that epipolar lines are parallel to the scanlines. These selected and processed data are then subsampled to 256 by 256 pixel for DFSS analysis. The geometry information are transformed into the aligned global coordinate system, as described in section 2.5. The down-sizing is done only because limited time and CPU power to process full resolution images. ¹

5.4 Data Analysis

In this section, I present two typical examples of Viking image pairs. The DFSS result are shown to demonstrate the effectiveness and robustness of *z-only* Depth from Shading and Stereo algorithm.

5.4.1 Example 1

The first pair of images is of an terrain surface similar to eroded hills and river bed. Figure 5-1 and Figure 5-2 shows shows the camera geometry and its projection in xz and yz plane. The light sources come from almost the same direction. Fortunately, the light source direction is very oblique and different from the camera direction results in much stronger shading information comparing with hard crater case. Table 5.1 lists the geometric information about this pair of images.

The Camera Angles are listed in the right ascension, declination, and twist of the camera. The Spacecraft Vector is the position of the spacecraft/camera in the reference frame of the planet. The Planet Angles are the orientation of the planet in the

¹At 256 by 256 pixel, it takes about 20 hours to finish 1200 iterations

	Camera Angle	Spacecraft Vector	Planet Angle	Sun Vector
Left	-68.449593	-1324.127686	52.694637	154312733.273824
	313.565796	1092.557129	317.313019	159262758.890330
	274.407928	5395.773438	225.757187	68965427.707157
Right	-63.648964	-1796.010010	52.694637	152618273.748284
	339.906433	853.954346	317.313019	160370845.199162
	297.853882	5504.281250	225.624329	69519512.685777

Table 5.1: Geometric Information about the first pair of Viking Images

same reference frame as the Camera Angles, also in right ascension, declination and twist. The Sun Vector is the position of the Sun in the planet reference frame. These information of orientation and position are converted to the geometric information in the global reference frame and shown in Figure 5-1 and Figure 5-2. Figure 5-3 shows the reflectance map contour and the image pair.

Figure 5-4 to Figure 5-7 shows the results of applying the DFSS algorithm to this pair of images. *z-only* DFSS algorithm performs well, correctly estimated the complicated topology. Obviously, considerable more iterations are required to obtain a satisfactory estimate of the surface than the synthetic case. Even so, the algorithm formed a stable estimation after about 500 iterations. Also, the cost function remains higher than that in synthetic image case.

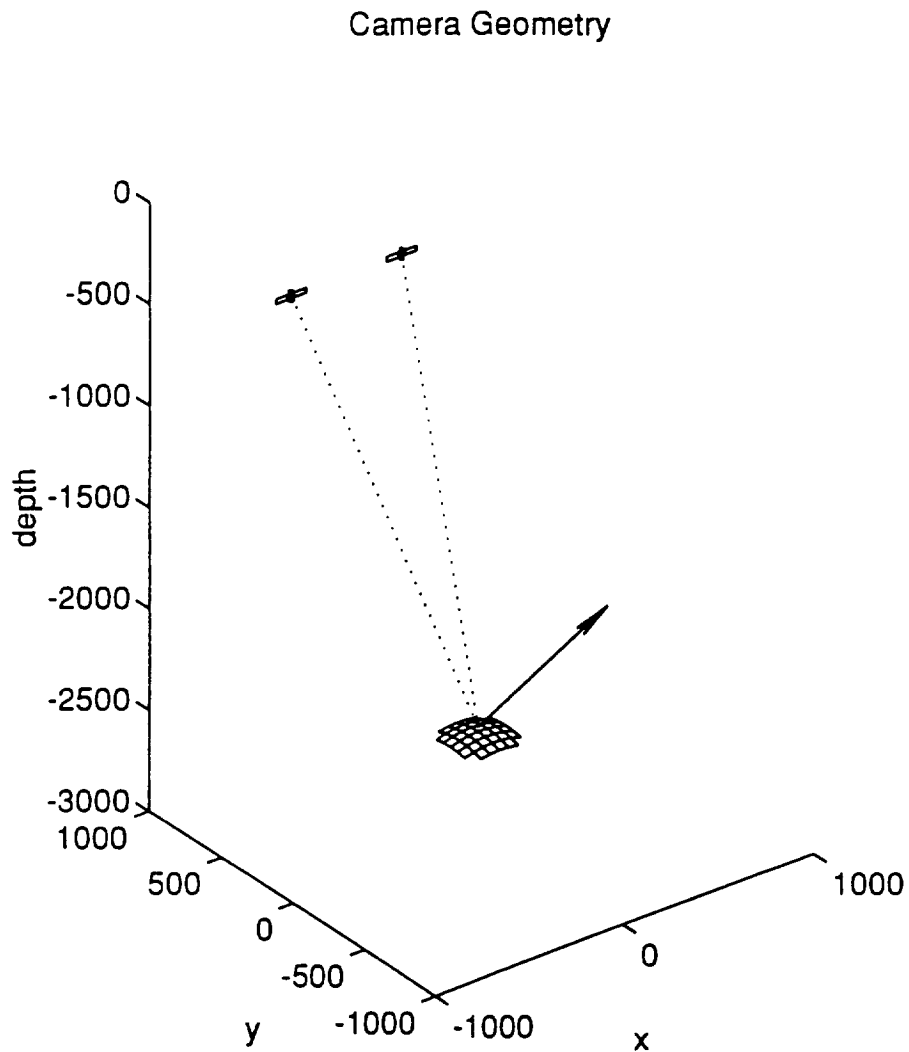


Figure 5-1: Mars Viking Images pair Camera Geometry. Graph shows the camera position and direction(dotted lines), as well as the light source direction(solid lines)

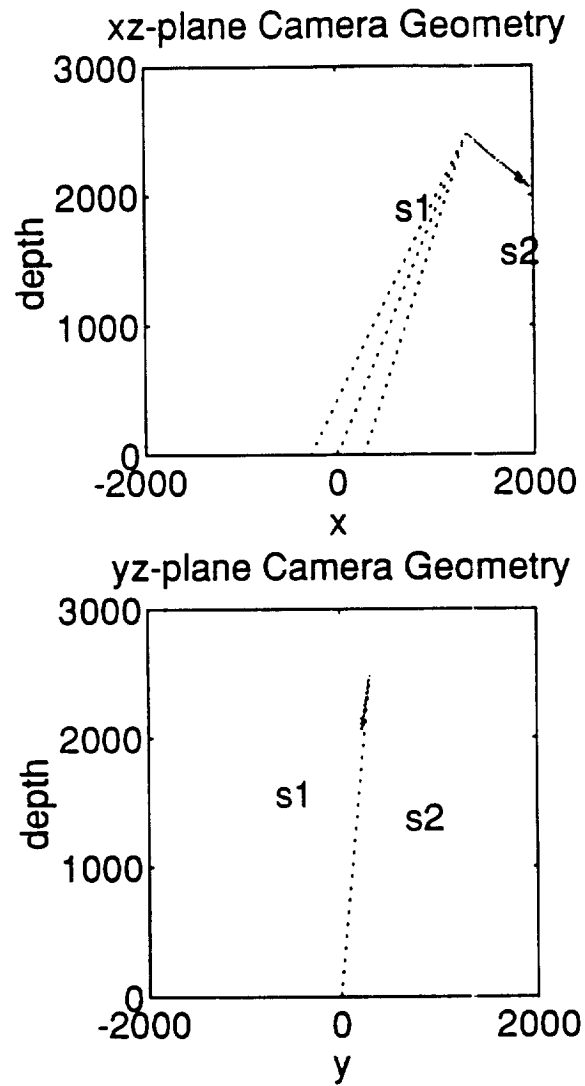
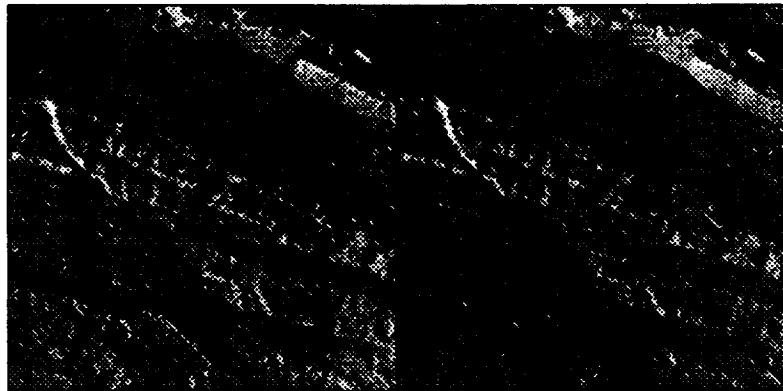
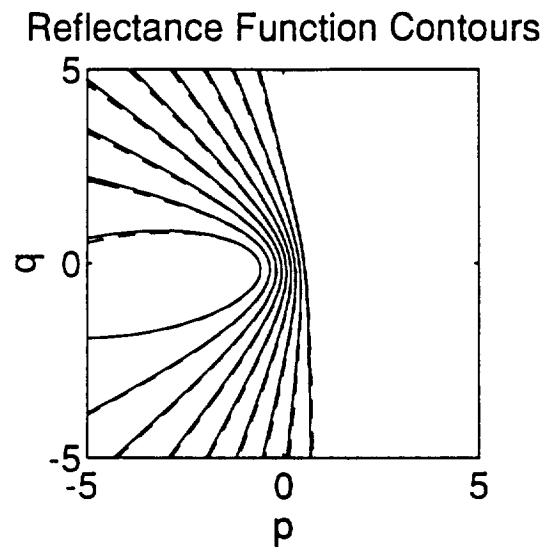


Figure 5-2: Mars Viking Images pair Camera Geometry projected in xz and yz plane



True images

Figure 5-3: Mars Viking images pair and gradient contour

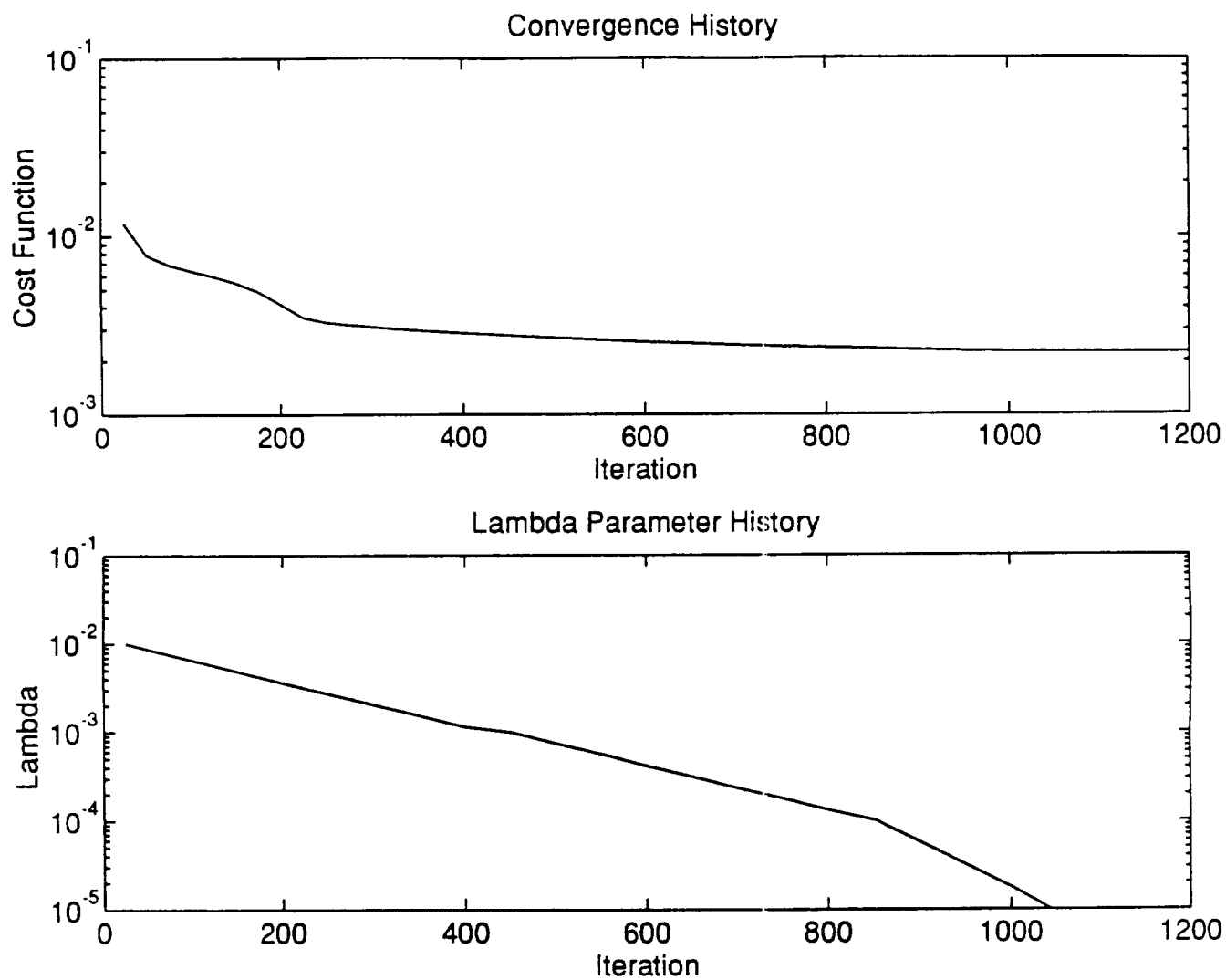


Figure 5-4: DFSS iteration history on Mars Viking Image pair. Top graph shows the history of cost function. Bottom graph shows the history of the smoothness weighting term λ

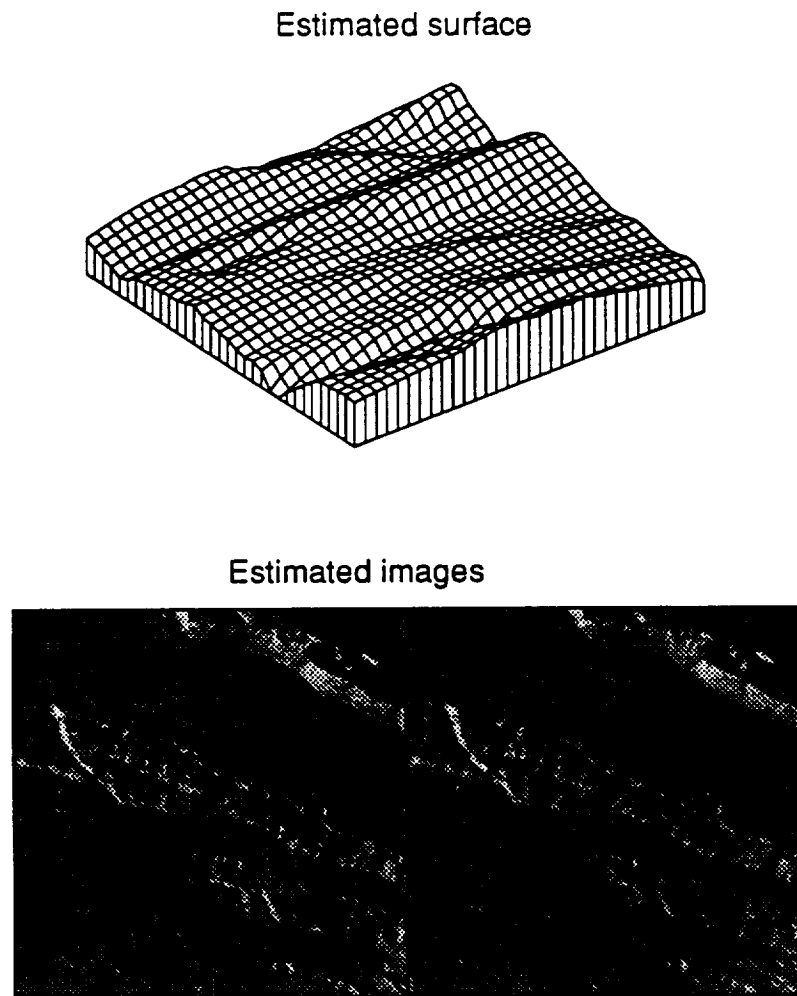


Figure 5-5: DFSS Estimated images and Surface on Mars Viking Image pair at the end of iterations

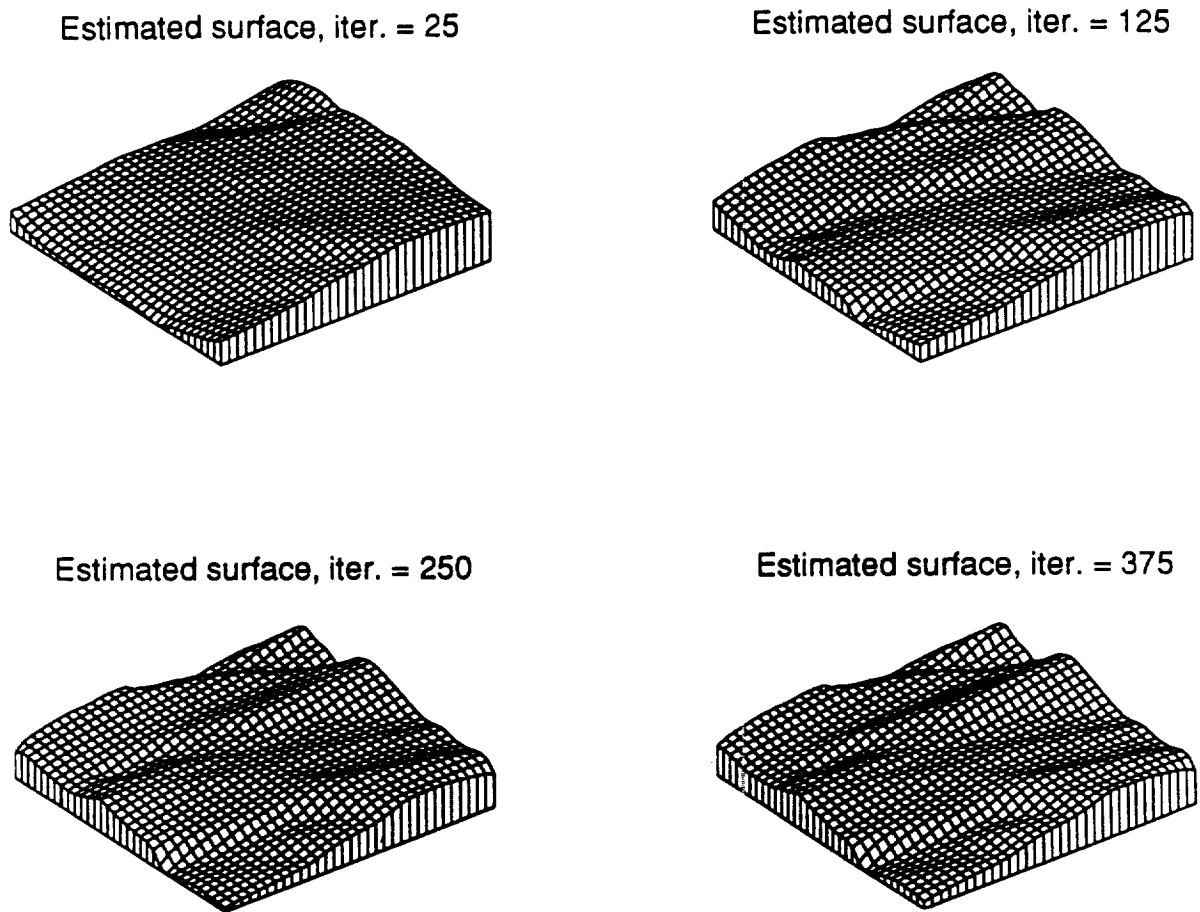
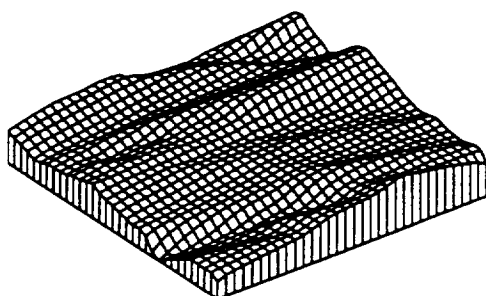
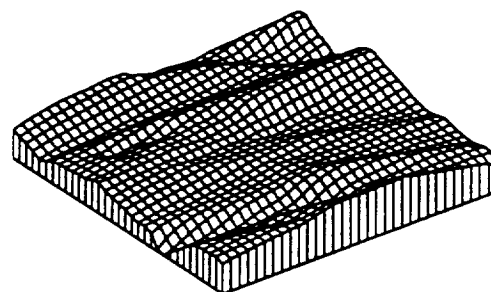


Figure 5-6: DFSS estimation at various iteration steps on Mars Viking Image pair

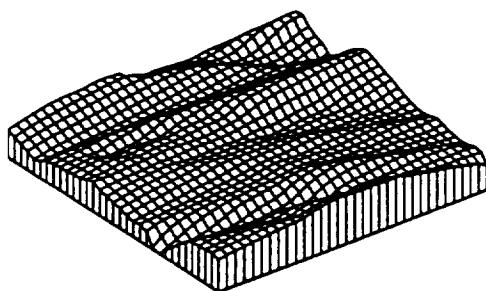
Estimated surface, iter. = 600



Estimated surface, iter. = 940



Estimated surface, iter. = 1100



Estimated surface, iter. = 1200

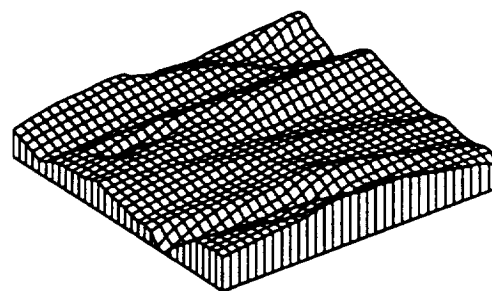


Figure 5-7: DFSS estimation at various iteration steps on Mars Viking Image pair

	Camera Angle	Spacecraft Vector	Planet Angle	Sun Vector
Left	-38.500999	3031.000000	52.694817	247003002.182980
	-114.101997	3252.000000	317.313354	-6496592.410365
	-153.830994	2154.000000	60.560833	-9607699.669818
Right	-16.581617	3431.279785	52.694775	241693638.758362
	221.748749	3228.368896	317.313263	37984763.457939
	250.999435	1721.439697	66.486511	10952544.914326

Table 5.2: Geometric Information about the first pair of Viking Images

5.4.2 Example 2

The second pair of images is also of an terrain surface similar to eroded hills and river bed. Figure 5-8 and Figure 5-9 shows the camera geometry and its projection in xz and yz plane. The light sources come from almost the same direction. Similar to example 1, there is little separation in the direction of the light sources. This also presents a challenge to the DFSS algorithm. The table below Table 5.2 lists the geometric information about this pair of images. In Figure 5-10, the reflectance map contour and the image pair are shown.

The Camera Angles are listed in the right ascension, declination, and twist of the camera. The Spacecraft Vector is the position of the spacecraft/camera in the reference frame of the planet. The Planet angles are the orientation of the planet in the same reference frame as the Camera Angles, also in right ascension, declination and twist. The Sun Vector is the position of the Sun in the planet reference frame. These information of orientation and position are converted to the geometric information in the global reference frame and shown below.

Figure 5-11 to Figure 5-14 shows the results of applying the DFSS algorithm to this pair of images. *z-only* DFSS algorithm performs reasonably well, closely estimated the complicated topology. Obviously, considerable more iterations are required to obtain a satisfactory estimate. Similarly to example 1, after 500 iterations, DFSS algorithm formed a stable estimation.

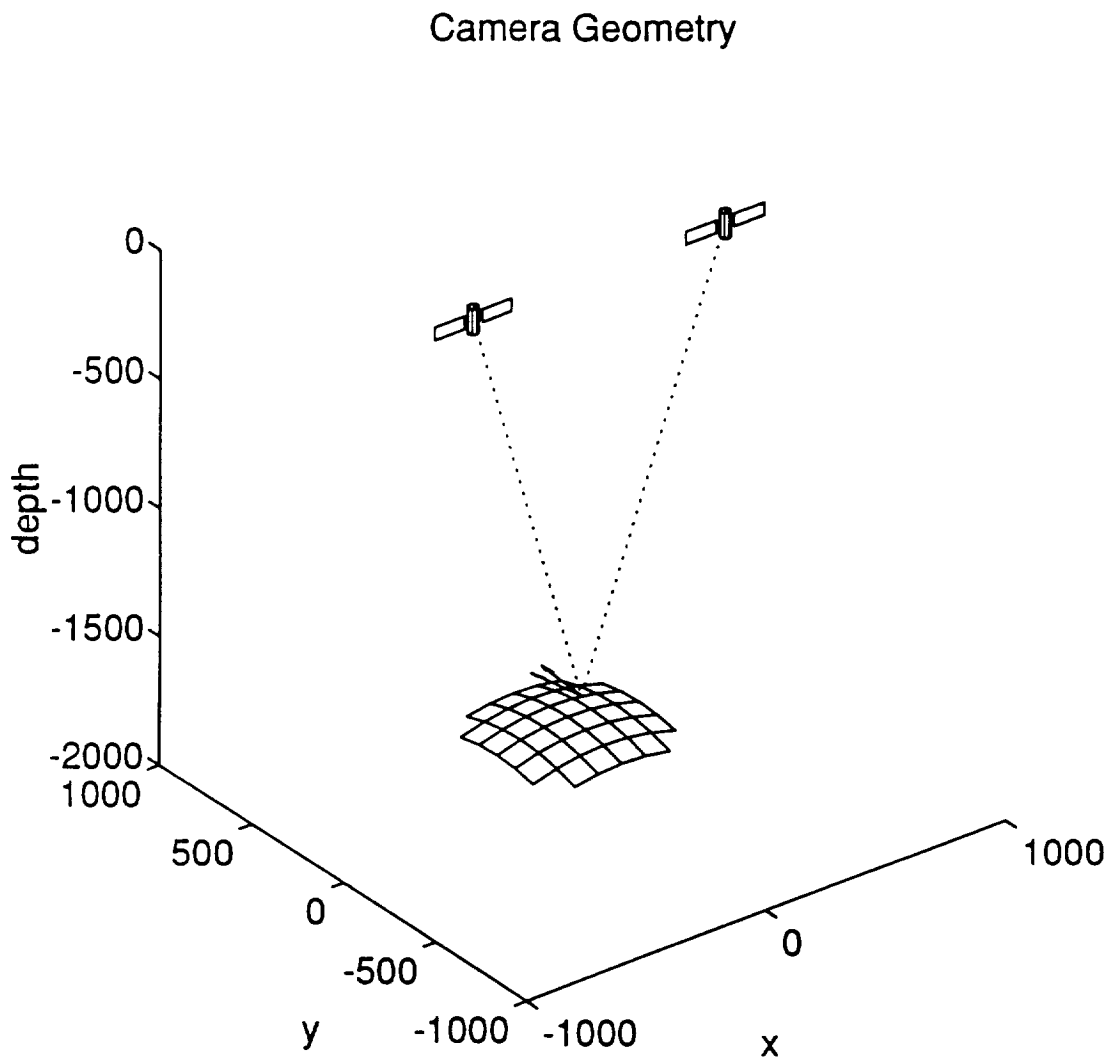


Figure 5-8: Mars Viking Images pair Camera Geometry. Graph shows the camera positions and directions(dotted lines), also the light source direction(solid lines).

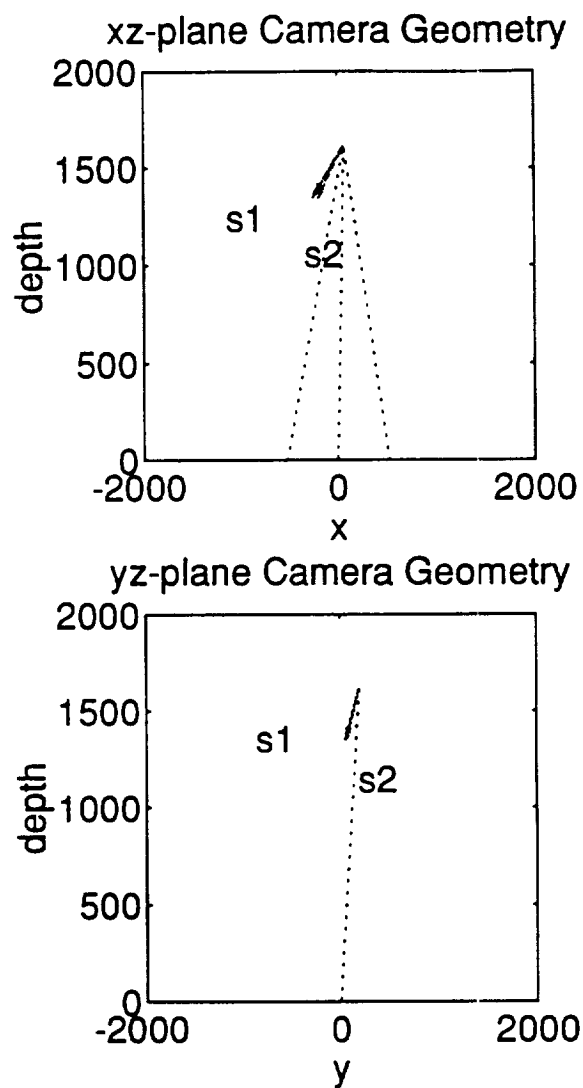
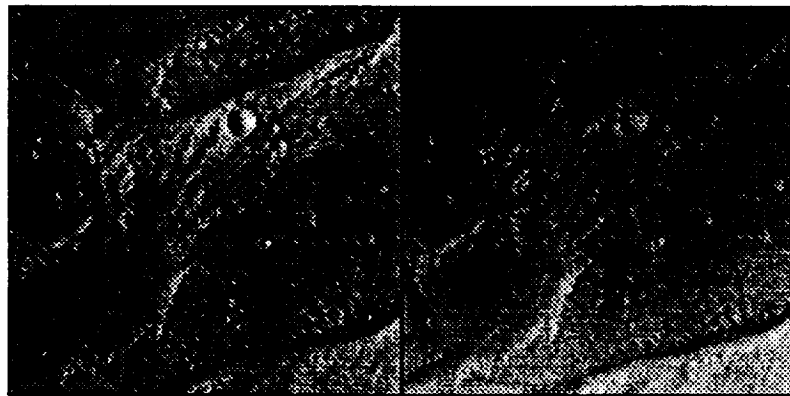
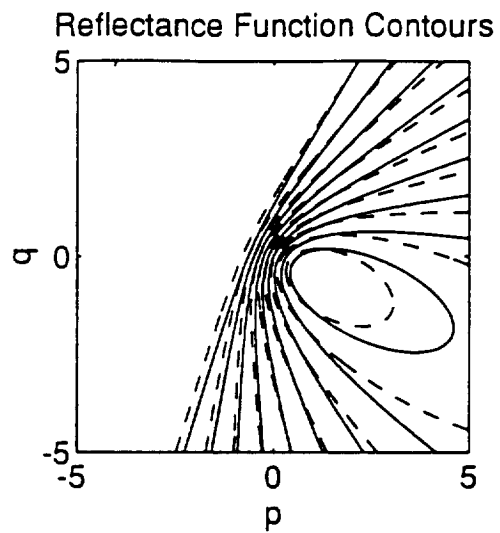


Figure 5-9: Mars Viking Images pair Camera Geometry projected in xz and yz plane



True images

Figure 5-10: Mars Viking Images pair and gradient contour

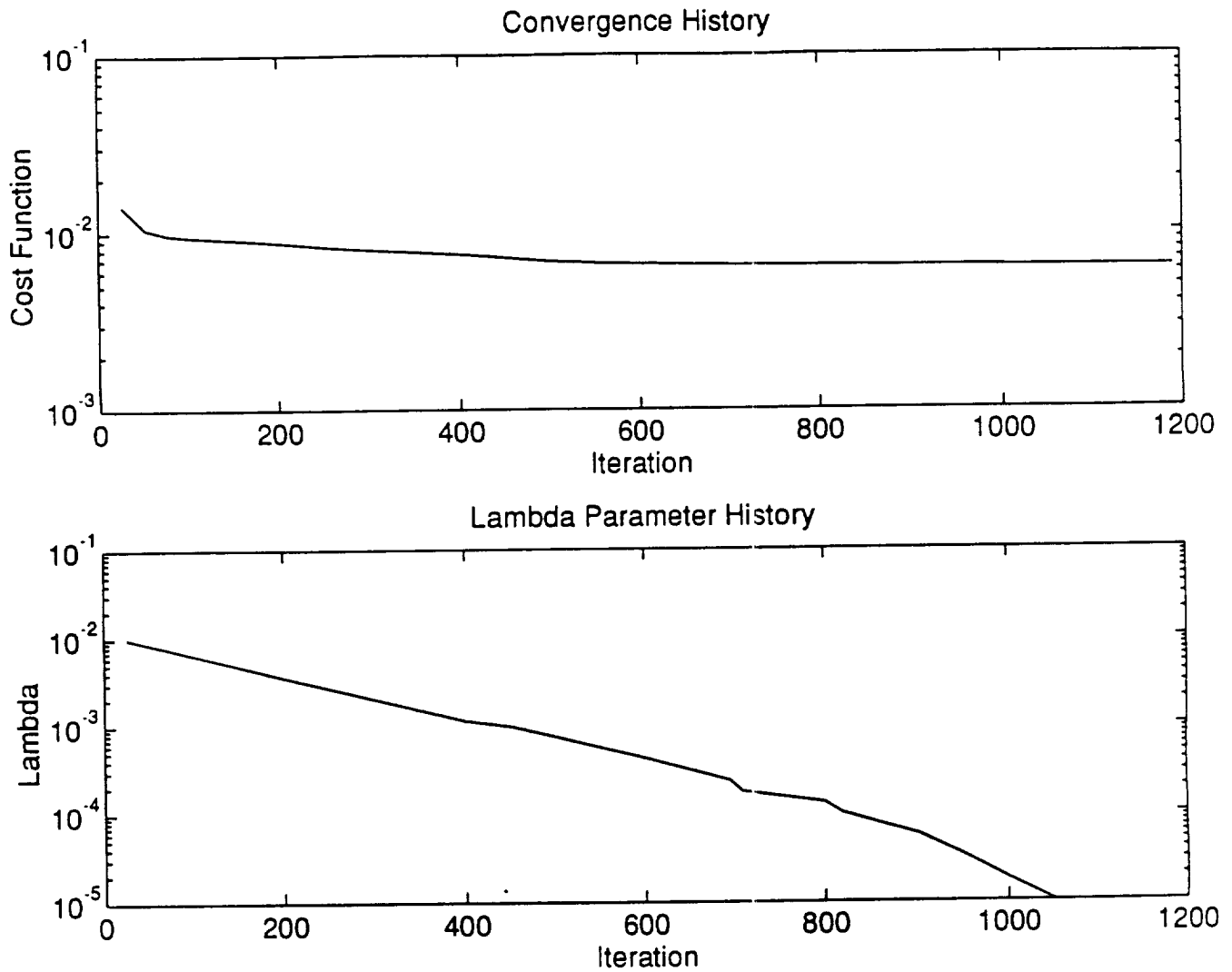
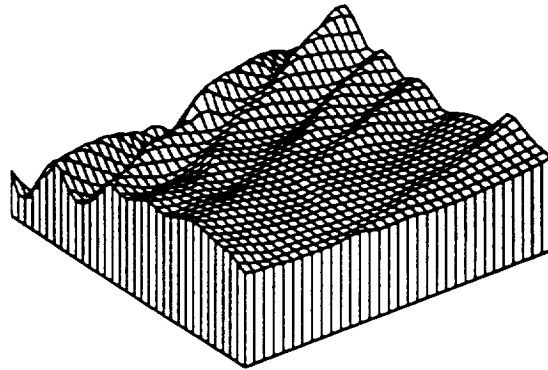


Figure 5-11: DFSS iteration history on Mars Viking Image pair. Top graph shows the history of cost function. Bottom graph shows the history of smoothness weighting term λ

Estimated surface



Estimated images

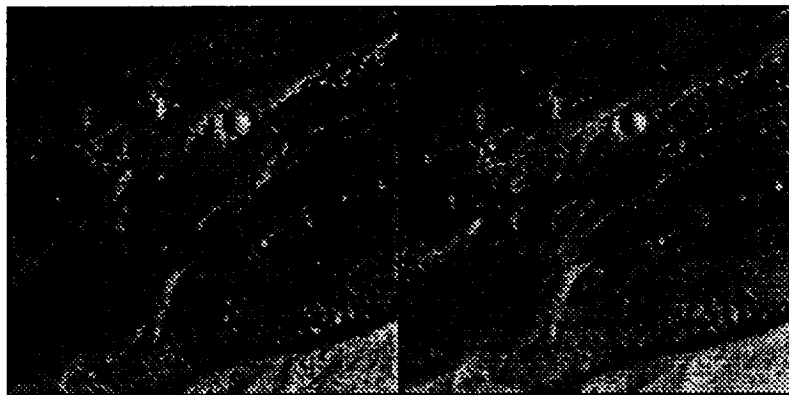
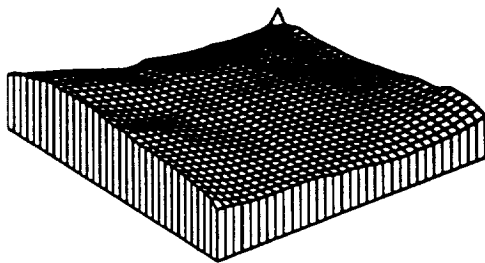
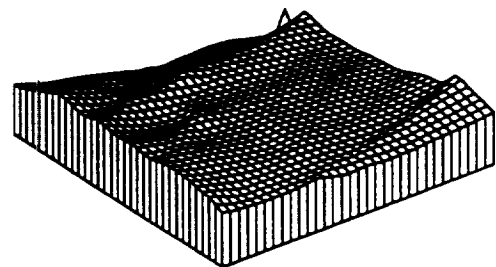


Figure 5-12: DFSS Estimated Surface on Mars Viking Image pair at the end of iterations

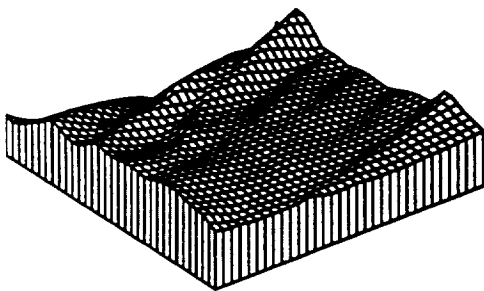
Estimated surface, iter. = 25



Estimated surface, iter. = 125



Estimated surface, iter. = 250



Estimated surface, iter. = 375

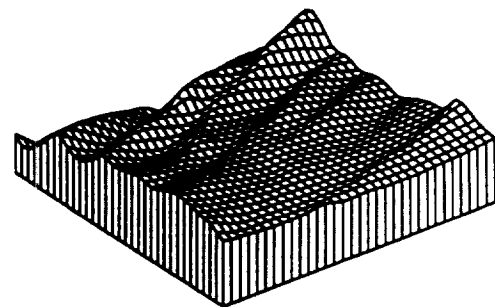
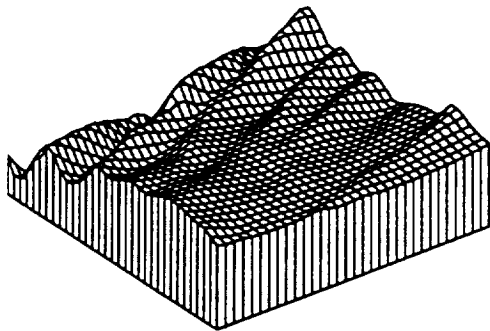
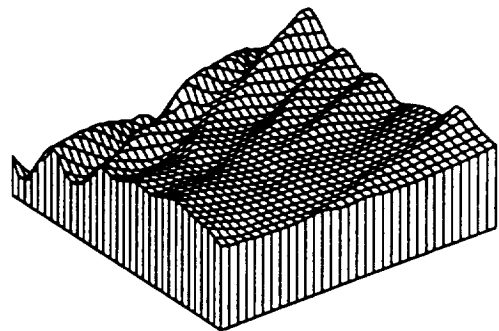


Figure 5-13: DFSS estimation at various iteration steps on Mars Viking Image pair

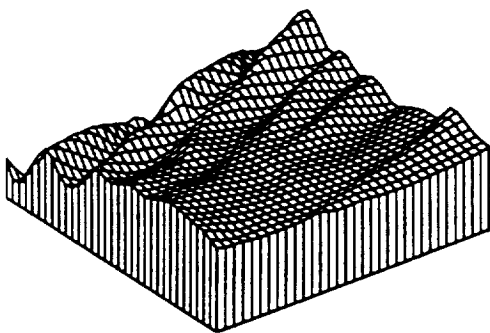
Estimated surface, iter. = 600



Estimated surface, iter. = 940



Estimated surface, iter. = 1100



Estimated surface, iter. = 1200

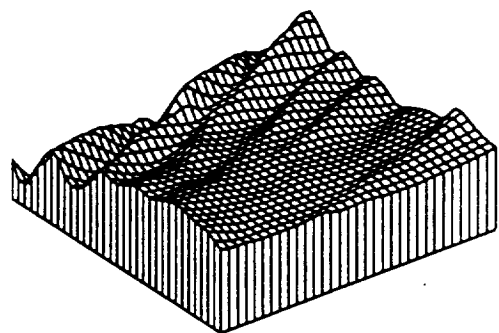


Figure 5-14: DFSS estimation at various iteration steps on Mars Viking Image pair

5.5 Summary of the Results

The results of *z-only* DFSS algorithms running on two example Mars Viking image pairs are presented in the previous sections. They've both demonstrated the adequate performance and robustness of this algorithm. Especially, the algorithm's performance under which the two light sources come in from the same or almost the same direction.

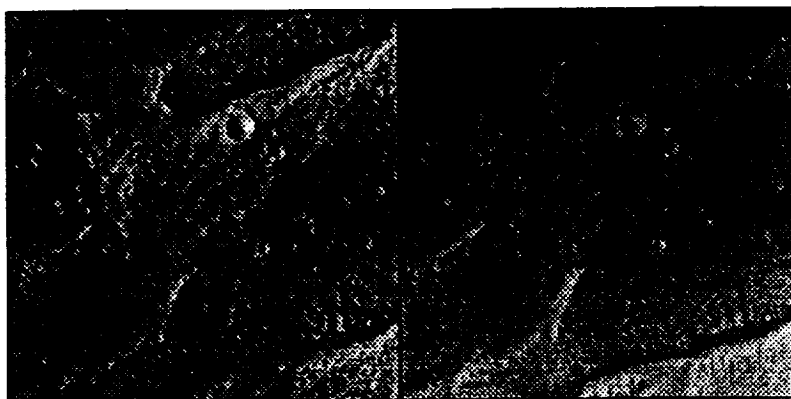
From the test results on the real Viking image pairs, clearly *z-only* DFSS can form close estimation on surface topology in a reasonable number of iterations. There is a general increase of the iterations needed comparing with synthetic image case. But after 500 iterations, the estimate is already quite stable and quite close to the true surface.

The result also shows that *z-only* algorithm is robust, able to estimate the topological surface in the presence of several types of errors.

One thing is not addressed clearly by the earlier two examples, that is one of the advantages of Shape from Shading is to be able to detect details of the shape. Comparing the estimated images and the real images, one can see most of the features are represented. In the mesh plot, however, it is not obvious. This is largely due to the coarseness of the mesh plot. To demonstrate that the details are there, we can zoom in on one particular feature. In this case, I chose the crater like feature in the second example.

Clearly the crater feature is vivid in the enlarged portion of the real image. Both the estimated image and surface present such a feature, however, a cluster of much smaller craters which are visible in real image is smeared and hardly recognizable in the estimated image and surface.

Full images



Zoomed in images around crater feature

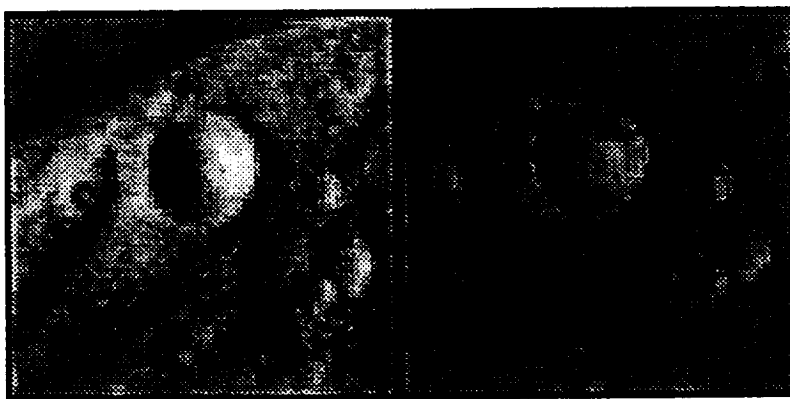
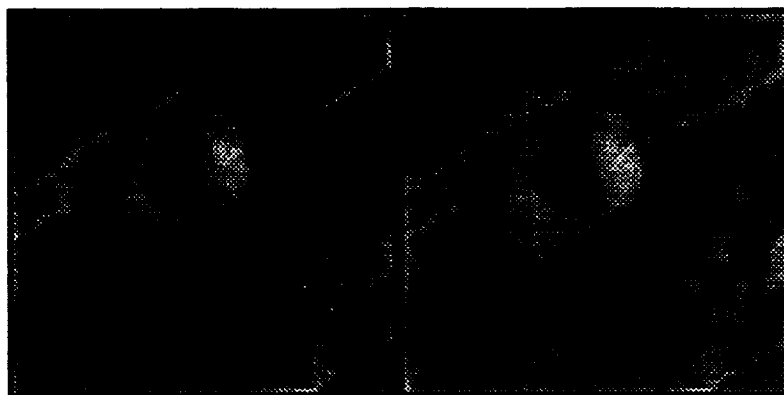


Figure 5-15: Detailed look of the crater like feature in example two, real images and zoomed in images around the crater.

Estimated images around crater feature



Estimated surface around crater feature

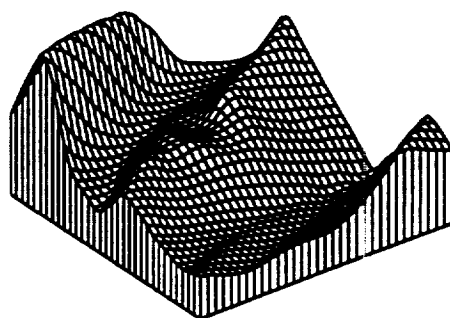


Figure 5-16: Detailed look of the crater like feature in example two, estimated images and surface around the crater.

Chapter 6

Error Analysis

It is of crucial importance to understand the various errors and assumptions associated with the images and the methods used to analysis them in earlier chapter.

6.1 Theoretical Error

It is important to review the assumptions behind those simplified equations. The perspective projection equations assume perfect lenses and perfect knowledge of the camera principal points and optical axes. The surface radiance equation assumes we have perfect knowledge of the surface reflectance properties, light source directions, and all surface points as visible from both cameras. The simplified form of the image irradiance equation assumes we either have a perfect sensor or we can perfectly calibrate the sensor to remove any abnormalities from the sensor & lens combination. The stereo equations assume we know the relative position and orientation of the two camera perfectly. the only assumption that are truly artificial are the assumptions of perfect knowledge of the reflectance maps and the the assumption of no self-illumination. With more careful measurements and more expensive equipment, it is possible to approach perfect knowledge of other assumptions. The assumption that all surface points be visible merely restricts the roughness of the surface that

this research is applicable to.

It is as well as important to go over the simplified equations and point out the their effects. There are several simplifications that have been made to the equation.

1. A special global coordinate system that is halfway between the two camera position.
2. All surface points are assumed to be visible from the two cameras.
3. The reflectance properties of the surface are assumed to be constant and *Lambertian* allowing the viewing direction dependence to be dropped from the reflectance equations. In addition, it is assumed that there is no interreflection between different parts of the surface.
4. The surface is assumed to have constant albedo allowing the position dependent term to be dropped.
5. The cameras optical axis are assumed to be aligned with each other allowing the rotational transformation to be set to identity.

With all the above simplifications, the set of equations are:

$$\begin{aligned} E^{(1)}(\mathbf{r}_1) &= R^{(1)}(\hat{\mathbf{n}}) \\ E^{(2)}(\mathbf{r}_2) &= R^{(2)}(\hat{\mathbf{n}}) \end{aligned} \tag{6.1}$$

where

$$\begin{aligned} \mathbf{r}_1 &= \frac{f(\xi + b/2)}{(\xi + b/2) \cdot \hat{\mathbf{z}}_1}, \\ \mathbf{r}_2 &= \frac{f(\xi - b/2)}{(\xi - b/2) \cdot \hat{\mathbf{z}}_2}, \end{aligned} \tag{6.2}$$

If we define $z = \xi \cdot \hat{z}_G$ and $\mathbf{r} = f\xi/z$, then the constraint function can be written as

$$\begin{aligned} E^{(1)}(\mathbf{r} + \frac{f\mathbf{b}}{2z}) &= R^{(1)}(\hat{\mathbf{n}}) \\ E^{(2)}(\mathbf{r} - \frac{f\mathbf{b}}{2z}) &= R^{(2)}(\hat{\mathbf{n}}) \end{aligned} \quad (6.3)$$

Use explicitly $\mathbf{r} = (x, y, f)$ and the normal vector $\hat{\mathbf{n}}$ is parameterized using gradient component p and q ,

$$\hat{\mathbf{n}} = \frac{(-p, -q, 1)}{\sqrt{p^2 + q^2 + 1}} \quad (6.4)$$

where

$$\begin{aligned} p &= \frac{f z_x}{x z_x + z} \\ q &= \frac{f z_y}{y z_y + z} \end{aligned} \quad (6.5)$$

The photo-topography equations then become

$$\begin{aligned} E^{(1)}(x + \frac{fb}{2z}, y) &= R^{(1)}(p, q) \\ E^{(2)}(x - \frac{fb}{2z}, y) &= R^{(2)}(p, q) \end{aligned} \quad (6.6)$$

In order to relate positions in the image direction vectors in 3-D space, the origin of the camera coordinate system must be known. Finding this origin is the classical *interior orientation* problem. In the special coordinate system we use here, see Figure 2-8, the position of each camera coordinate system origin can be specified by a vector $\mathbf{v} = (v_x, v_y, f^T)$. these vector specify the offset of the camera coordinate origin

for each image. Suppose \mathbf{v}_0 is the offset to an object point in the global coordinate system, then the offset to this same point in the camera images are

$$\mathbf{v}_1 = \mathbf{v}_0 + \frac{fb}{2z_0}, \quad (6.7)$$

$$\mathbf{v}_2 = \mathbf{v}_0 - \frac{fb}{2z_0}. \quad (6.8)$$

the values of \mathbf{v}_1 and \mathbf{v}_2 can be quite large in the aligned coordinate system indicating that the images must be shifted far away from the camera coordinate system origin. While this is not possible physically, it is a consequence of re-projecting real images into the aligned coordinate system.

In removing viewing direction dependency, we use orthographic projection for the reflectance function while retaining perspective projection for everything else. This introduced an error into the calculations since orthographic and perspective projection do not produce the same viewing direction in general. This error is so small for planet photo-topography, since the cameras are so far away from the surface. The large viewing distance results a very small relative change in depth. This is especially true here due to the small viewing angle.

Also, this algorithm only deals with uniformly colored surface, in another word, no varying albedo. This assumption is quite valid in the case we are discussing.

The above simplified equations helped formulate the algorithm and produce reasonable result, it's also hidden some important information, such as the normal vector is parameterized using gradient components p and q , instead of vector. Also it imposed restrictions such as all the surface point are visible from two cameras.

6.2 Image Errors

As discussed in earlier chapter, the images taken during Viking Project consisted of many error, noise and needs calibration and corrections.

Many Viking Orbiter images are missing data and contain some amount of noise. a common pattern of missing data is a series of vertical bars with zero value pixel spaced at an interval of 7 samples. In addition, data for a few horizontal image lines may be missing and such lines are filled with zero values.

The noise found in these images include single-pixel random noise and several source of coherent noise. The random noise is usually due to telemetry errors. The coherent noise arises from shuttering of the adjacent camera, filter wheel stepping, and scan platform movements. The coherent noise typically exists in the top or bottom 100 lines of an image and appearing as a "herring bone" pattern. Box filtering techniques that fill in zero values or average the bright and dark spikes of random noise are often successful and used.

As described in chapter 5, the Viking images are radiometrically calibrated by converting the digital signal received from the camera to a quantity that is proportional to the radiance reaching the sensor.

6.3 Lighting Condition, Camera Position and the effects on DFSS results

More important here is the effects of the geometry information under which the images are taken. To illustrate the effects, we regenerate the synthetic easy crater case, under similar lighting, camera geometry and baseline condition of to that of the real images, in particular example 1 in Chapter 5.

Similar to the set used in chapter 4, these images are based on a crater on a flat surface. It is very simple, and thus serves a good test bed. As shown in Figure 6-1,

comparing this pair with the pair in chapter 4, contour plots of the reflectance map as well as true images clearly show the effect of the change of lighting. Examining the estimated images and surface, apparently *z-only* DFSS algorithm estimated this test crater surface. But notice the flat background is distorted, which means under this kind of lighting condition, a small anomaly can affect the estimated surface and the programs' convergence. The DFSS results is shown in Figure 6-4.

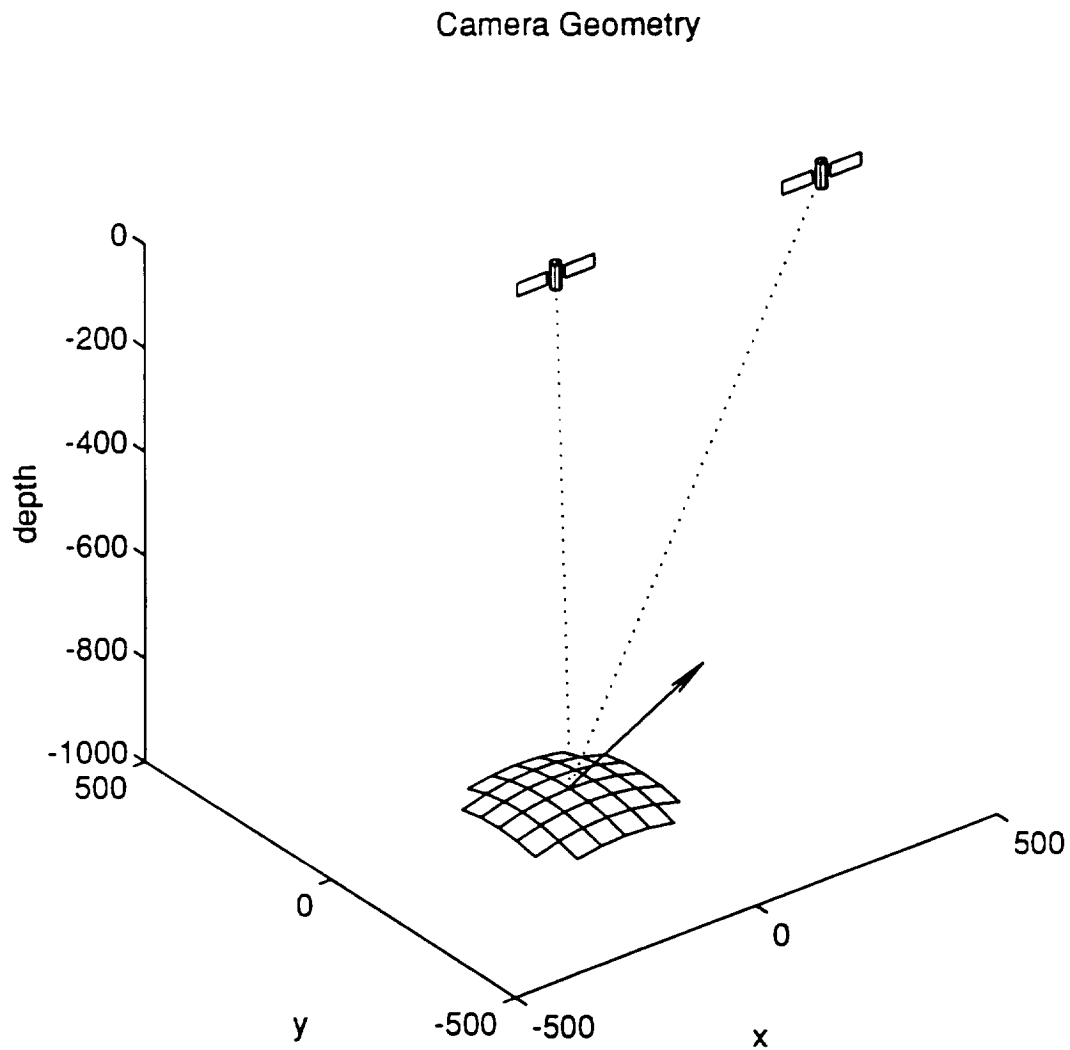


Figure 6-1: Test Crater Synthetic Images Camera Geometry. Graph shows the camera position and direction(dotted lines) and light source direction(solid lines)

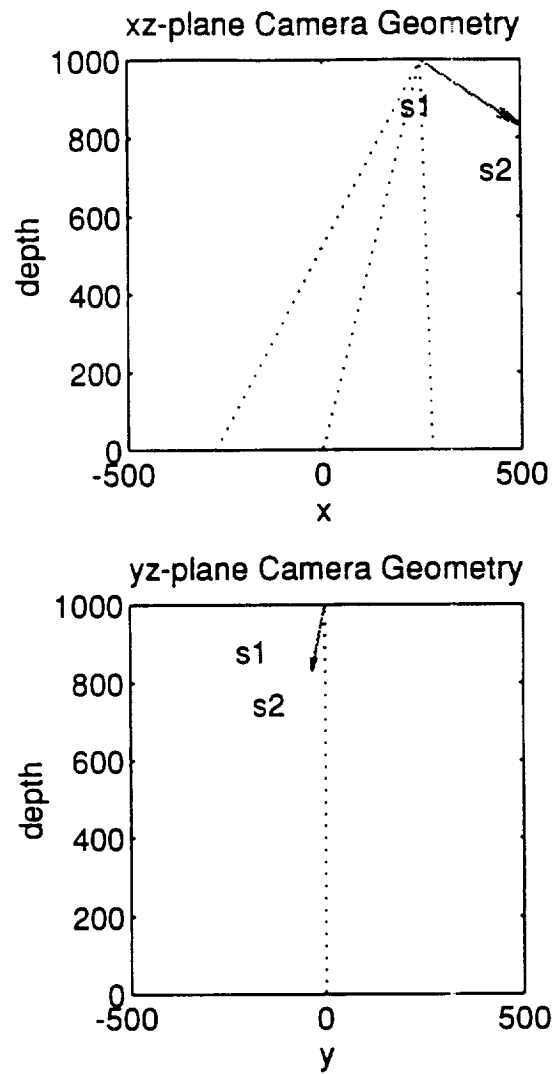
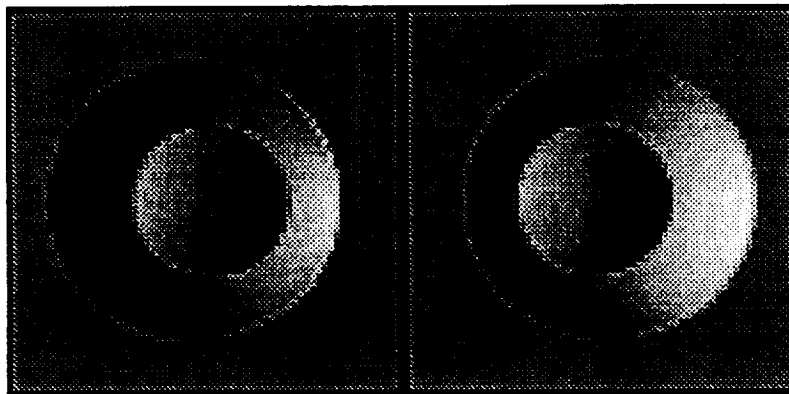
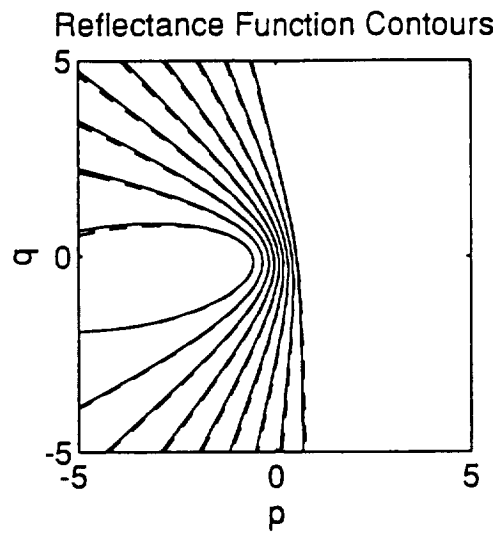


Figure 6-2: Test Crater Synthetic Images Camera Geometry projected in xz and yz plane



True images

Figure 6-3: Test Crater Synthetic Images

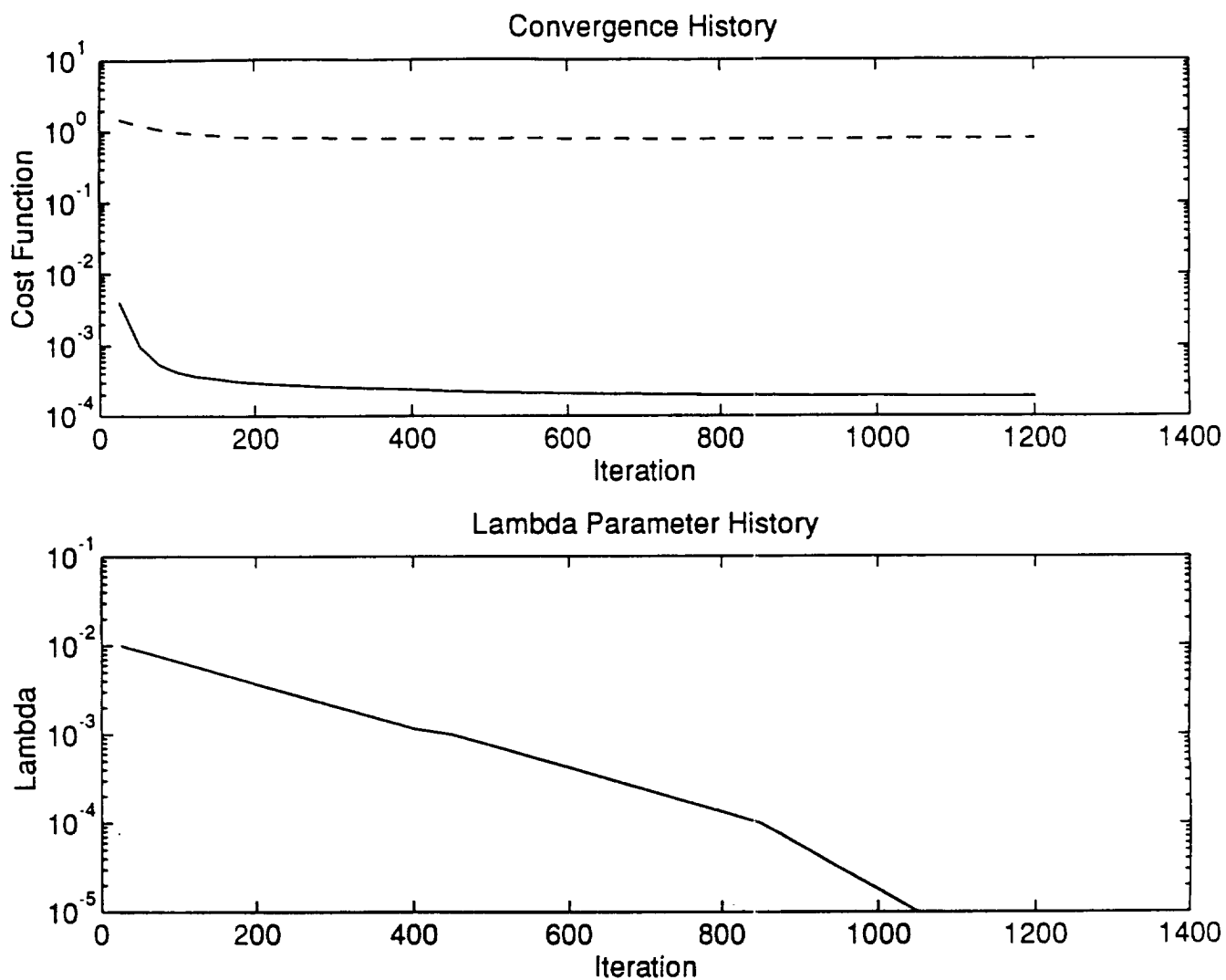
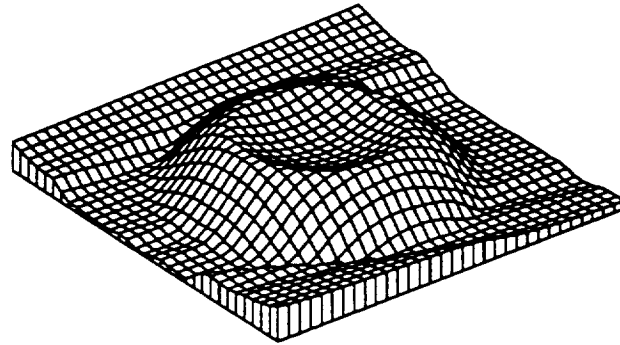


Figure 6-4: *z-only* DFSS iteration history on Test Crater Images. Graph shows the *z-only* DFSS algorithm performance on the test crater image pair. The upper graph shows the cost function(solid line) and RMS error(dashed line) of the estimated surface. The lower graph shows history of the smoothness weighting term λ

Estimated surface



Estimated images

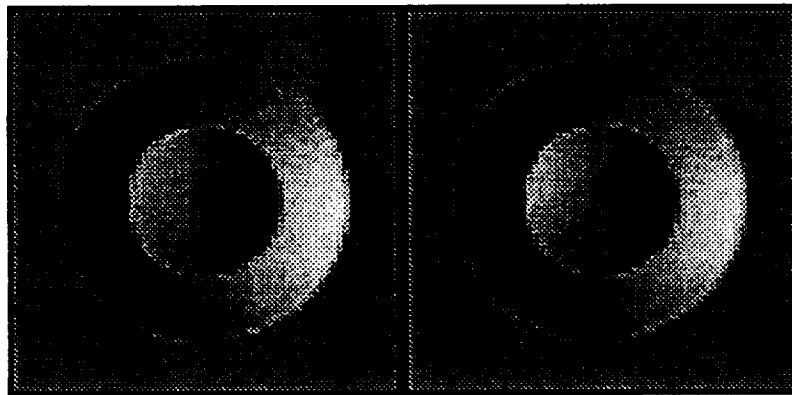
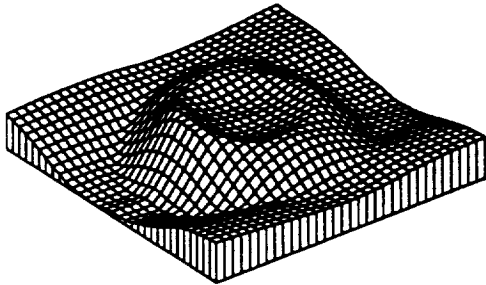
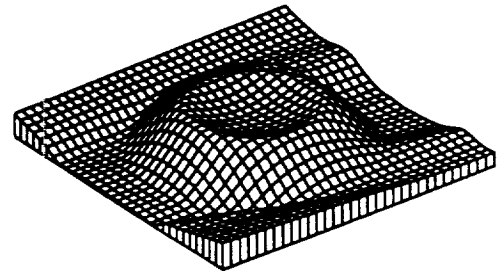


Figure 6-5: *z-only* DFSS Estimated Surface and Error on Test Crater Images at the end of iterations

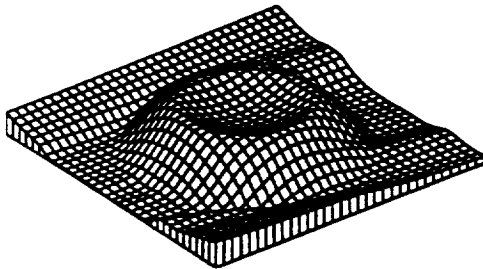
Estimated surface, iter. = 25



Estimated surface, iter. = 125



Estimated surface, iter. = 250



Estimated surface, iter. = 375

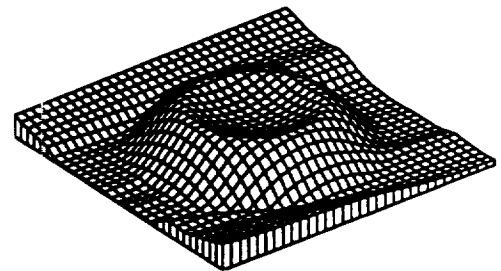
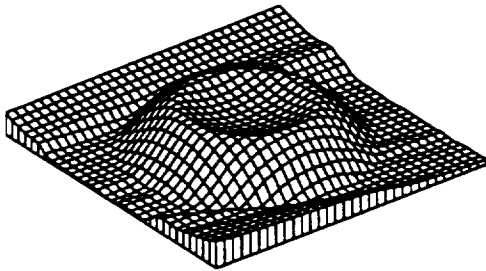
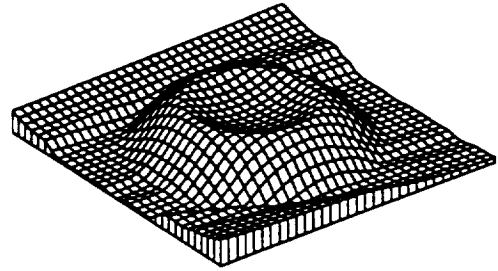


Figure 6-6: *z-only* DFSS estimation at various iteration steps on Test Crater Images

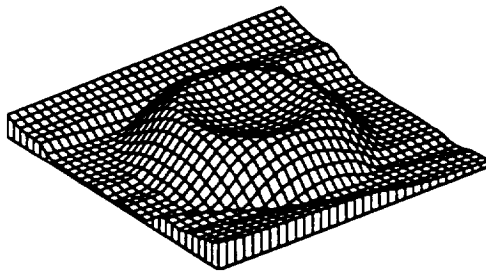
Estimated surface, iter. = 600



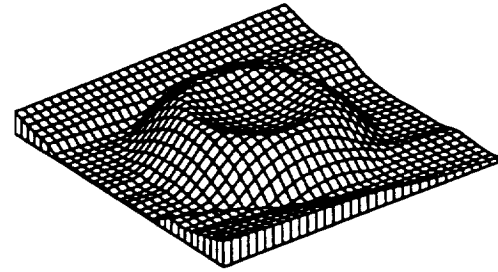
Estimated surface, iter. = 940



Estimated surface, iter. = 1100



Estimated surface, iter. = 1200

Figure 6-7: *z-only* DFSS estimation at various iteration steps on Test Crater Images

6.4 Comparison between DFSS results and that of Shape from Shading

The earlier two examples presented in chapter 5 prompt us two questions. The first one is that the two images are taken under similar lighting and camera position, the two images look alike except slight change in shading, therefore what kind result the Shape from Shading algorithm will provide? The second is that from photometric stereo, we know that the more different the lighting condition is, the better the result is. In this way, we can regard Shape from Shading as photometric stereo under exactly the same lighting condition. DFSS algorithm on the other hand has integrated photometric stereo into itself. So can DFSS algorithm produce better result?

These two questions are actually the same, they can be rephrased as “is DFSS algorithm working better than Shape from Shading in dealing the real images we used here, and why?”. The answer is yes. The reason is that DFSS fused photometric stereo, binocular stereo and shape from shading. It can take advantage of the difference in shading and gradient in the image pairs. This lends another handle to constrain the estimated surface, particularly at the estimated of overall trend of the surface, where Shape from Shading is weak. Even though the differences are small in the two examples in this thesis. The benefit is obvious.

To show and prove the advantage of DFSS, we can go back to previous section, and run one image through Shape from Shading. The reason to do so is that those images pairs are synthetically generated according to the same lighting and geometry information of that of example 1 in chapter 5. This is better and easier than the real images because they are simple structure and we know the ground truth.

Clearly, Shape from shading closely estimated the surface. Comparing the estimated surface with that from DFSS, DFSS predicted a much flat background surface and the crater feature is better shown.

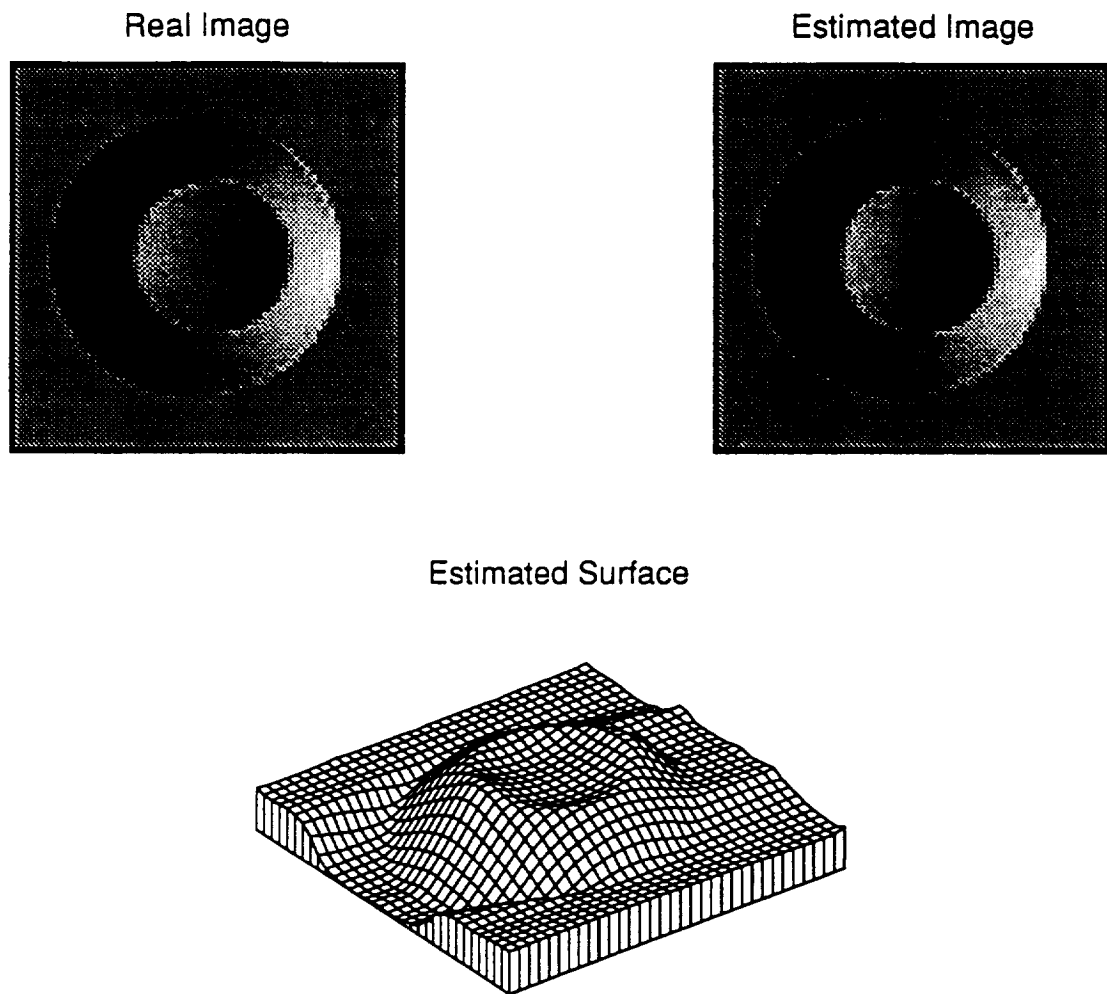


Figure 6-8: Test Crater Synthetic Image (left camera) and the estimated surface at the end of iteration.

Chapter 7

Discussion

7.1 Merits

z – *only* algorithms demonstrate a much better performance using the two photogrammetry images than a simple shape from shading algorithm which uses only one image. This performance increase validates the fusion approach to obtaining better performance vision algorithms.

The z – *only* algorithm was also shown to be robust, it is able to accurately estimate the synthetic surface in the presence of several types of errors. The performance of the algorithm based on images that contains noise, geometry error, or reflectance errors was demonstrated in real Mars images. In which case, the algorithm is able to form a close estimate.

The z – *only* DFSS algorithm shows very good performance on real Mars Viking image pairs. Especially it shows its performance under real measurement errors, calibrations error, distortion, and noise. It also shows its robustness under relatively weak shading information.

The C implementation of z – *only* DFSS algorithm makes it more portable and efficient. Better Than three times the performance is obtained than its version in matlab. Large image(256x256) can now be processed on a SPARCstation.

7.2 limitations

There are many limitations in the Depth from Shape from Shading and Stereo algorithm.

Though z -only algorithm is the most robust one among the proposed zpq , Dual- z and Disparity map. It does not handle varying albedo, nor does it handle very weak shading situation very well. Along these, there are some technical deficiencies, such as reflectance map depends on scalar p and q , rather than vector P and Q .

DFSS has strong bias towards Shape-from-Shading algorithm. Since it is variational method based, easy to integrate when shading information is weak, DFSS algorithm can get stuck in a local minimum, as seen in hard crater case. The stereo information presented in both the synthetic and real images are strong and can be used to pull the algorithm out of the local minimum. However since binocular stereo is feature based, makes it difficult to incorporate into the cost function.

In dealing with real images, it is found that DFSS algorithm performs not very satisfactory when lighting are coming from similar direction. Though this is not DFSS was developed for, it must also be able to cope with real situations.

The DFSS algorithm is based on simplified geometry, It works the best when the distance from the object to the two cameras are roughly equal. This is not always true or easily obtainable in the real world. This also contributes to the difference of the performance between the real and the synthetic images.

There is also an implementation issue. For 256 by 256 pixel image, the program runs roughly 20 hours to finish 1200 iteration with maximum number of hierarchical functions. To process full resolution images, 1204 by 1056 pixel, is not practical.

Since the thesis' main goal is to test z -only DFSS algorithm on real Viking Images images. I am not going to discuss about many extensions possible to this algorithm. But any extension which may result significant memory and CPU time requirement increase, such as the write reflectance maps as a function of vector P and Q . Instead focus should be on improving optimization process.

Appendix A

DFSS Functions

In this Appendix, I will briefly explain all the DFSS core processing functions and their relationship. From these functions, one can build a program to process and analysis any images fit DFSS feature, as discussed in earlier chapters, by providing proper I/O functions.

The source code of the following functions and a makefile are made available on <ftp.ai.mit.edu>(128.52.32.11), in /pub/yan, along with a copy of this thesis.

The program loops over the conjugate gradient optimization function. In the conjugate gradient optimization calculation. It first calculates the cost based on current estimates of the surface. The gradient function returns the gradient associated with it. Base on these, a linear search is performed to find a new minimum, and the cost and gradient are then recalculated to use this new value. This continues until the predetermined number of iteration run out. During this process the hierarchical basis conversion functions are used to speed up calculation. Interpolation, filtering and convolution functions are used to assist the cost and gradient calculation. Reflection map based on current estimates are generated and compared with the actual reflection map to guide the convergence.

dfss.h	header file of DFSS functions
dfss.c	main DFSS analysis function
conjgrad.c	conjugate gradient calculation
cost.c	cost calculation
grad.c	gradient calculation
hbasis.c	hierarchical basis conversion
lsearch.c	linear search for conjugate gradient optimization
conv2.c	2-D convolution calculation
cfilter2d.c	2-D computational molecule filtering with bicubic interpolation
domain2d.c	2-D plaid domain generation
interpx.c	linear interpolation in one direction.
rmap.c	reflection map calculation based on current estimates.

Table A.1: DFSS function list

Appendix B

DFSS I/O Functions

In this Appendix, I present several example I/O and driver functions needed to build a DFSS program to analyze and process Viking Space project images. Since I used Matlab to display and plot the images and the estimated surface, loadmat.c and savemat.c are used to read and write Matlab files.

The source code and a makefile are made available on ftp.ai.mit.edu(128.52.32.11), in /pub/yan, along with a copy of this thesis.

```
/* *****  
 * FILE dfss_pds.c *  
 * *  
 * Depth From Shape from Shading and Stereo Image Analysis Program *  
 * for PC, Vax, Unix and Macintosh systems. *  
 * *  
 * it reads synthetic images in matlab file format and output *  
 * estimated topography in matlab file format. *  
 ***** */
```

```
#include <stdio.h>  
#include <math.h>  
#include <malloc.h>  
#include <strings.h>
```

```

#include <matrix.h>
#include "dfss.h"

void main(int argc, char **argv)
{
    char                name[20];
    int                 mz, nz, type, mrows, ncols, imagf;
    real                *xi, *xr, *x;
    matrix               ztemp;

    /* set up global variables */

    levels = 7;          /* level of hbasis */

    cntl1 = 1.0e-8;       /* Termination tolerance for X. */
    cntl2 = 1.0e-16;      /* Termination tolerance on F. */
    cntl3 = 1.0e-6;       /* Termination criterion on constraint violation. */
    cntl4 = 0;            /* Algorithm: Line Search Algorithm. */
    cntl5 = 0;            /* Function value. (Lambda in goal attainment) */
    cntl6 = 0;            /* Number of Function and Constraint Evaluations. */
    cntl7 = 0;            /* Number of Function Gradient Evaluations. */
    cntl8 = 0;            /* Number of Constraint Evaluations */
    cntl9 = 0;            /* Number of equality constraints. */
    cntl10 = 0;           /* Maximum number of iterations. default, 100 * no of var. */
    cntl11 = 1.0e-8;      /* Min. change in variables for finite diff. grad. */
    cntl12 = 0.1;         /* Max. change in variables for finite diff. grad. */
    cntl13 = 0.0;         /* Step length. (Default 1 or less) */
    cntl14 = 0.0;

    /* get host information and input and output files */

    strcpy(inname1, " ");
    strcpy(outname, " ");

```

```

if (argc == 1)
    usage();
else if (argc == 2 && (strcmp(argv[1], "-help", 5) == 0 ||
    strcmp(argv[1], "-h", 2) == 0 ||
    strcmp(argv[1], "?", 1) == 0))
    usage();
else {
    strcpy(inname1, argv[1]);
    if (argc == 3) strcpy(outname, argv[2]);
    if (argc > 3) usage();
}

host = check_host();
get_files(host);

/* readin the image file */

printf("\nReading in the reflection maps ..... \n");

while (loadmat(infp1, &type, name, &mrows, &ncols, &imagf, &xr, &xi) == 0) {
    if (strcmp(name, "E1", 2) == 0) {
        e1 = new_matrix(mrows, ncols);
        array_to_matrix(e1, mrows, ncols, xr);
    }
    if (strcmp(name, "E2", 2) == 0) {
        e2 = new_matrix(mrows, ncols);
        array_to_matrix(e2, mrows, ncols, xr);
    }
    if (strcmp(name, "z", 1) == 0) {
        mz = mrows; nz = ncols;
        ztemp = new_matrix(mrows, ncols);
        array_to_matrix(ztemp, mrows, ncols, xr);
    }
    if (strcmp(name, "params", 6) == 0) {

```

```

    f = *xr; b = *(xr + 1); z0 = *(xr + 2);
    ps1 = *(xr + 3); qs1 = *(xr + 4); ps2 = *(xr + 5); qs2 = *(xr + 6);
    vx1 = *(xr + 7); vy1 = *(xr + 8); vx2 = *(xr + 9); vy2 = *(xr + 10);
    delta = *(xr + 11);
}
}

```

```

ztrue = new_matrix(mz - 2, nz - 2);           /* strip off the boarder */
matrix_copy_submatrix(ztemp, ztrue, 1, mz - 1, 1, nz - 1);
free_matrix(ztemp);

```

```

x = calloc((mz - 2) * (nz - 2), sizeof(double));
matrix_to_array(ztrue, mz - 2, nz - 2, x);
savemat(outfp, host, "ztrue", mz - 2, nz - 2, 0, x, xi);
free(x);

```

```

free_matrix(ztrue);

```

```

z = new_matrix(mz - 2, nz - 2);           /* allocate memory for z */
matrix_set(z, z0);

```

```

/* start DFSS analysis */

```

```

dfss();

```

```

/* output the final depth matrix */

```

```

x = calloc(NOROWS(e1) * NOCOLS(e1), sizeof(double));
matrix_to_array(e1, NOROWS(e1) , NOCOLS(e1), x);
savemat(outfp, host, "e1", NOROWS(e1) , NOCOLS(e1) 0, x, xi);
free(x);

```

```

x = calloc(NOROWS(e2) * NOCOLS(e2), sizeof(double));
matrix_to_array(e2, NOROWS(e2) , NOCOLS(e2), x);

```

```
savemat(outfp, host, "e2", NOROWS(e2) , NOCOLS(e2), 0, x, xi);
free(x);
```

```
x = calloc((mz - 2)* (nz - 2), sizeof(double));
matrix_to_array(z, mz - 2 , nz - 2, x);
savemat(outfp, host, "z", mz - 2 , nz - 2, 0, x, xi);
free(x);
```

120

```
/* clear up and close */
```

```
free_matrix(z);
free_matrix(ztrue);
free_matrix(e1);
free_matrix(e2);
```

130

```
close(infp1);
fclose(outfp);
}
```

```
/* subroutine get_files - get input, output filenames and open them */
```

```
void get_files(int host)
{
    if (inname1[0] == ' ') {
        printf("\nEnter name of the file to be DFSS analyzed: ");
        gets (inname1);
    }
    if (host == 0 | host == 3000) {
        if ((infp1 = fopen(inname1, "rb")) == NULL) {
            printf("\ncan't open input file: %s\n", inname1);
            exit(1);
        }
    }
}
```

140

150

```

else if (host == 2000 | host == 1000) {
    if ((infp1 = fopen(inname1, "r")) == NULL) {
        printf("\ncan't open input file:  %s\n", inname1);
        exit(1);
    }
}

```

/ get output file */*

```

if (outname[0] == ' ') {
    printf("\nEnter name of output file:  ");
    gets (outname);
}
if (host == 0 | host == 2000) {
    if ((outfp = fopen(outname, "wb")) == NULL) {
        printf("\ncan't open output file:  %s\n", outname);
        exit(1);
    }
}
else if (host == 1000) {
    if ((outfp = fopen(outname, "w")) == NULL) {
        printf("\ncan't open output file:  %s\n", outname);
        exit(1);
    }
}
}

```

160

170

```

void usage(void)
{
    printf("z-only Depth From Shading and Stereo Program\n\n");
    printf("INPUT:\t synthetic image in matlab file format.\n");
    printf("OUTPUT:\t results in matlab file format\n");
    printf("\nCommand line format:\n\n");
    printf("dfss_mat [infile] [outfile] \n");
}

```

180

```

printf("\tinfile\t - name of synthetic image file.\n");
printf("\toutfile\t - name of DFSS output file.\n");
}

```

```

/*****
 * FILE gen_opt.c
 *
 * make lambda and step table, used in optimization function
 *****/

```

```

#include <stdio.h>
#include <math.h>
#include <matrix.h>
#include "dfss.h"

```

10

```

matrix gen_opt(real *lambda, int *step, int *nsteps, int tot)
{
    int      i, j;                /* temp. counter */
    int      r_opt = 0;           /* count number of elements in opts */
    matrix    opt;                /* lambda and step matrix */
    matrix    steps, a, b, c;     /* temp matrixs */

    opt = new_matrix(tot, 2);     /* allocate memory for opt matrix */
    for (i = 0; i < 5; i++) {
        steps = new_matrix(1, nsteps[i]);
        for (j = 0; j < nsteps[i]; j++) {
            elem_set(steps, 0, j, (real) j+1);
        }
        a = new_matrix(nsteps[i], 1);
        matrix_set(a, log(lambda[i]));
        matrix_scalar_sub(steps, 1.0, steps);
        b = new_matrix(nsteps[i], 1);
        matrix_transpose(steps, b);
    }
}

```

20

```

matrix_scalar_div(b, (real) nsteps[i], b);
c = new_matrix(nsteps[i], 1);
matrix_set(c, log(lambda[i+1]) - log(lambda[i]));
matrix_mul(b, c, c);
matrix_add(a, c, c);

/* generate opt matrix */

for (j = 0; j < nsteps[i]; j++) {
    elem_set(opt, r_opt + j, 0, exp(elem_ref(c, j, 0)));
    elem_set(opt, r_opt + j, 1, (real) step[i]);
}
r_opt += nsteps[i];
}

/* clean up the temp matrix, release memory */

free_matrix(steps);
free_matrix(a);
free_matrix(b);
free_matrix(c);
return(opt);
}

```

```

/ *****
* FILE get_host.c
*
* Check host computer type,
* for PC, Vax, Unix and Macintosh systems.
*
* it finds out the machine type and byte swap
*****/

```



```
#include <stdio.h>
#include <strings.h>
#include <matrix.h>
#include "dfss.h"

int check_host()
{
    char      hostname[80];
    int       swap, host, bits, var;
    union {
        char  ichar[2];
        short ilen;
    } onion;

    if (sizeof(var) == 4) bits = 32;
    else bits = 16;

    onion.ichar[0] = 1;
    onion.ichar[1] = 0;

    if (onion.ilen == 1) swap = TRUE;
    else                swap = FALSE;

    if (bits == 16 && swap == TRUE) {
        host = 0; /* IBM PC host */
        strcpy(hostname,
            "Host 1 - 16 bit integers with swapping, no var len support.");
    }

    if (bits == 16 && swap == FALSE) {
        host = 0; /* Non byte swapped 16 bit host */
        strcpy(hostname,
            "Host 2 - 16 bit integers without swapping, no var len support.");
    }
}
```

```

if (bits == 32 && swap == TRUE) {
    host = 2000; /* VAX host with var length support */
    strcpy(hostname,
        "Host 3 - 32 bit integers with swapping.");
}

if (bits == 32 && swap == FALSE) {
    host = 1000; /* OTHER 32-bit host, such as Sun */
    strcpy(hostname,
        "Host 5 - 32 bit integers without swapping, no var len support.");
}

return(host);
}



---




---


/*****
* FILE loadmat.
*
* C language routine to load a matrix from a MAT-file.
*****/

#include <stdio.h>

typedef struct {
    long type; /* type */
    long mrows; /* row dimension */
    long ncols; /* column dimension */
    long imagf; /* flag indicating imag part */
    long namlen; /* name length (including NULL) */
} Fmatrix;

int loadmat(FILE *fp, int *type, char *pname, int *mrows, int *ncols,

```

```
int *imagf, double **preal, double **pimag)
{
    char *malloc();
    Fmatrix x;
    int mn, namlen;

    /* Get Fmatrix structure from file */

    if (fread((char *)&x, sizeof(Fmatrix), 1, fp) != 1) {
        return(1);
    }
    *type = x.type;
    *mrows = x.mrows;
    *ncols = x.ncols;
    *imagf = x.imagf;
    namlen = x.namlen;
    mn = x.mrows * x.ncols;

    /* Get matrix name from file */

    if (fread(pname, sizeof(char), namlen, fp) != namlen) {
        return(1);
    }

    /* Get Real part of matrix from file */

    if (!(*preal = (double *)malloc(mn*sizeof(double)))) {
        printf("\nError: Variable too big to load\n");
        return(1);
    }
    if (fread(*preal, sizeof(double), mn, fp) != mn) {
        free(*preal);
        return(1);
    }
}
```

```
/* Get Imag part of matrix from file, if it exists */
```

```

if (x.imagf) {
    if (!(*pimag = (double *)malloc(mn*sizeof(double)))) {
        printf("\nError: Variable too big to load\n");
        free(*preal);
        return(1);
    }
    if (fread(*pimag, sizeof(double), mn, fp) != mn) {
        free(*pimag);
        free(*preal);
        return(1);
    }
}
return(0);
}

```

60

```

/*****
* FILE savemat.c
* C language routine to save a matrix in a MAT-file.
*****/

```

```
#include <stdio.h>
```

```

typedef struct {
    long type; /* type */
    long mrows; /* row dimension */
    long ncols; /* column dimension */
    long imagf; /* flag indicating imag part */
    long namlen; /* name length (including NULL) */
} Fmatrix;

```

10

```
void savemat(FILE *fp, int type, char *pname, int mrows, int ncols,  
             int imagf, double *preal, double *pimag)  
{  
    Fmatrix x;  
    int mn;  
  
    x.type = type;  
    x.mrows = mrows;  
    x.ncols = ncols;  
    x.imagf = imagf;  
    x.namlen = strlen(pname) + 1;  
    mn = x.mrows * x.ncols;  
  
    fwrite(&x, sizeof(Fmatrix), 1, fp);  
    fwrite(pname, sizeof(char), (int)x.namlen, fp);  
    fwrite(preal, sizeof(double), mn, fp);  
    if (imagf) {  
        fwrite(pimag, sizeof(double), mn, fp);  
    }  
}
```

20

30

Appendix C

The Matrix Library

C.1 Vectors and Matrices

In the following section, I provide some documentation on the functions available in the matrix library for doing vector and matrix manipulation. The documentation covers only the basic functions.

The entire library is written in the C programming language and has been running on Sparc workstations in the AI LAB at MIT, and PCs outside.

In what follows, the library functions deal exclusively with 2-dimensional matrices and 1-dimensional vectors and do NOT deal with multi-dimensional matrices.

Internally, matrices are represented as two dimensional matrices. A vector of dimension n is represented as a column vector or an matrix whose shape is $n \times 1$.

The entire library has been written using a `typedef` which defines the `real` data type to be the C `double` datatype. By redefining this datatype in `matrix.h` and recompiling it should be relatively easy to convert the library to single precision if needed. To use the library the file `matrix.h` must be included in your C source files. The library defines (typedefs) many data types the most used of which is the `matrix` data type.

The source code and a makefile are made available on <ftp.ai.mit.edu>(128.52.32.11),

in /pub/yan, along with a copy of this thesis.

C.2 Creating and Freeing Matrices

The following function creates and returns an matrix all of whose elements are initialized to zero.

new_matrix	<i>rows, cols</i>	<i>Function</i>
<i>int rows, cols;</i>		

This is the routine used internally by all the others. This creates and returns an matrix with the specified number of rows and columns.

free_matrix	<i>a</i>	<i>Function</i>
<i>matrix a;</i>		

This routine performs the cleaning up, once you no longer need an matrix. It basically frees up the storage associated with the specified matrix a . Future references to a freed matrix will result in erroneous behaviour.

C.3 Accessing Elements within an Matrix

There are many functions to access elements within an matrix, the following two of which are very useful.

elem_set	<i>a, i, j, val</i>	<i>Function</i>
<i>matrix a;</i>		
<i>int i, j;</i>		
<i>real val;</i>		

It sets element referred to by `a[i,j]` to the real val.

elem_ref *a, i, j* *Function*

matrix a;

int i, j;

This returns the element referred to by *a[i,j]*.

These two functions are implemented as C functions and hence may be slow for some applications. If speed is essential, users can use macros **felem_set** and **felem_ref** with identical arguments as those of **elem_set** and **elem_ref**.

C.4 Common Operations on Matrices

matrix_add *a, b, c* *Function*

matrix a, b, c;

This adds two matrices together, if they are of the same shape. If specified result matrix *c* is NULL then the results are stored in matrix *b*.

matrix_sub *a, b, c* *Function*

matrix a, b, c;

Subtracts matrix *b* from *a*, if they are of the same shape. If specified result matrix *c* is NULL then the results are stored in matrix *b*.

matrix_mul *a, b, c* *Function*

matrix a, b, c;

Multiplies matrix *b* from *a*, if they are of the same shape. If specified result matrix *c* is NULL then the results are stored in matrix *b*. This does NOT do a matrix multiply. It does something like $c[i] = a[i] * b[i]$.

matrix_scalar_add *a, scalarvalue, b* *Function*

matrix a, b;

real scalarvalue;

This adds a scalar value to every element of the specified matrix **a**. If specified result matrix **b** is NULL, then the result is stored in matrix **a**;

matrix_scalar_sub *a, scalarvalue, b* *Function*

matrix a, b;

real scalarvalue;

This subtracts a scalar value from every element of the specified matrix **a**. If specified result matrix **b** is NULL, then the result is stored in matrix **a**;

matrix_scalar_mul *a, scalarvalue, b* *Function*

matrix a, b;

real scalarvalue;

This multiplies every element of the specified matrix **a** by the specified scalar value. If specified result matrix **b** is NULL, then the result is stored in matrix **a**;

matrix_scalar_div *a, scalarvalue, b* *Function*

matrix a, b;

real scalarvalue;

This divides every element of the specified matrix **a** by the specified scalar value. If specified result matrix **b** is NULL, then the result is stored in matrix **a**;

matrix_scalar_shift_add *a, scalar, b* *Function*

matrix a, b;

real scalarvalue;

This routine is like **matrix_scalar_add** only it shifts the matrix **b** by the specified amount instead of clobbering it with the result. In short doing a

```
matrix_scalar_shift_add(a, 3.0, b);
```

is equivalent to writing (in pseudo-code)

```
for all_elements_of b
    b[i] = b[i] + a[i] + 3.0;
```

Both matrices should be of the same shape.

matrix_scalar_shift_sub *a, scalar, b* *Function*

matrix a, b;
real scalarvalue;

This shifts the elements of matrix *a*, down instead of up as the add routine does.

matrix_scalar_shift_mul *a, scalar, b* *Function*

matrix a, b;
real scalarvalue;

This is equivalent to the following:

```
for all_elements_of b
    b[i] = b[i] + a[i] * scalar;
```

matrix_scalar_shift_div *a, scalar, b* *Function*

matrix a, b;
real scalarvalue;

This is equivalent to the following:

```
for all_elements_of b
    b[i] = b[i] + a[i] / scalar;
```

matrix_eq_internal *x,y,tol* *Function*

matrix x,y;

real tol;

This function checks if two matrices are equal on an element by element basis. The third argument specifies a tolerance value that is to be used in the comparison. It returns 1 if the two matrices are equal within the specified tolerance and 0 if not.

matrix_eq *x,y* *Function*

matrix x,y;

This is really equal to a function call to the previous function with a tolerance value of 0.00001.

matrix_norm_one *a* *Function*

matrix a;

Computes the maximum column sum of the specified matrix (also known as the 1-norm).

matrix_norm_infinity *a* *Function*

matrix a;

Computes the maximum row sum of the specified matrix. (also known as the inf-norm).

matrix_norm_frobenius *a* *Function*

matrix a;

Computes and returns the frobenius norm of the given matrix.

matrix_minmax *a, minval, maxval* *Function*

matrix a;

*real *minval;*

*real *maxval;*

This function computes and returns the maximum value and minimum value stored in the matrix in the result pointers passed in as the second and third arguments.

matrix_max *a, row, col*

Function

*register matrix a;**int *row, *col;*

This function returns the maximum value in the matrix and sets the passed in *row* and *col* arguments to be the row and column index of the element that corresponds to this maximum value.

matrix_average *a*

Function

register matrix a;

This function returns the average value of all the elements in an matrix.

matrix_multiply *a, b, c*

Function

matrix a, b, c;

This is the normal matrix multiply routine. The result matrix *c* must be specified, and must be of the right dimensions.

matrix_multiply_new *a, b*

Function

matrix a, b;

Returns a new matrix which is the result of matrix *a* post multiplied by *b*. It is upto the programmer to free the storage allocated to this matrix.

matrix_multiply_destructive *a, b*

Function

matrix a, b;

This routine destructively modifies the matrix *b* to contain *a* post-multiplied by *b*.

matrix_invert *a, b*

Function

matrix a, b; This is the familiar matrix inversion routine. Handles only square matrices now. The inverse of *a* is stored in *b*.

matrix_multiple_solve *a, b, c*

Function

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188	
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.				
1. AGENCY USE ONLY (Leave blank)		2. REPORT DATE November 1994		3. REPORT TYPE AND DATES COVERED Contractor Report
4. TITLE AND SUBTITLE Topography from Shading and Stereo			5. FUNDING NUMBERS 930	
6. AUTHOR(S) Berthold Horn				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Artificial Intelligence Laboratory Massachusetts Institute of Technology Cambridge, MA 02139			8. PERFORMING ORGANIZATION REPORT NUMBER NAS5-32337 5555-10 NAS5-31352	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) National Aeronautics and Space Administration - OSSA Washington, D.C. 20546-0001 Universities Space Research Association 10227 Wincopin Circle, Suite 212 Columbia, MD 21044			10. SPONSORING/MONITORING AGENCY REPORT NUMBER CR-189389	
11. SUPPLEMENTARY NOTES Technical Monitor: J. Hollis, Code 930				
12a. DISTRIBUTION/AVAILABILITY STATEMENT Unclassified-Unlimited Subject Category 82 Report is available from the NASA Center for AeroSpace Information, 800 Elkridge Landing Road, Linthicum Heights, MD 21090; (301) 621-0390.			12b. DISTRIBUTION CODE	
13. ABSTRACT (Maximum 200 words) Methods exploiting photometric information in images that have been developed in machine vision can be applied to planetary imagery. Integrating shape from shading, binocular stereo and photometric stereo yields a robust system for recovering detailed surface shape and surface reflectance information. Such a system is useful in producing quantitative information from the vast volume of imagery being received, as well as in helping visualize the underlying surface.				
14. SUBJECT TERMS Machine vision, phototopography, computer vision			15. NUMBER OF PAGES 189	
			16. PRICE CODE	
17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT Unlimited	

